THE UNIVERSITY *of* EDINBURGH

Neuromorphic Systems for Legged Robot Control

Hugo Alexandre Pereira Monteiro

THE UNIVERSITY
of EDINBURGH

A thesis submitted for the degree of Doctor of Philosophy.
The University of Edinburgh.
2013

Abstract

Locomotion automation is a very challenging and complex problem to solve. Besides the obvious navigation problems, there are also problems regarding the environment in which navigation has to be performed. Terrains with obstacles such as rocks, steps or high inclinations, among others, pose serious difficulties to normal wheeled vehicles. The flexibility of legged locomotion is ideal for these types of terrains but this alternate form of locomotion brings with it its own challenges to be solved, caused by the high number of degrees of freedom inherent to it.

This problem is usually computationally intensive, so an alternative, using simple and hardware amenable bio-inspired systems, was studied. The goal of this thesis was to investigate if using a biologically inspired learning algorithm, integrated in a fully biologically inspired system, can improve its performance on irregular terrain by adapting its gait to deal with obstacles in its path.

At first, two different versions of a learning algorithm based on unsupervised reinforcement learning were developed and evaluated. These systems worked by correlating different events and using them to adjust the behaviour of the system so that it predicts difficult situations and adapts to them beforehand. The difference between these versions was the implementation of a mechanism that allowed for some correlations to be forgotten and suppressed by stronger ones.

Secondly, a depth from motion system was tested with unsatisfactory results. The source of the problems are analysed and discussed. An alternative system based on stereo vision was implemented, together with an obstacle detection system based on neuron and synaptic models. It is shown that this system is able to detect obstacles in the path of the robot.

After the individual systems were completed, they were integrated together and the system performance was evaluated in a series of 3D simulations using various scenarios. These simulations allowed to conclude that both learning systems were able to adapt to simple scenarios but only the one capable of forgetting past correlations was able to adjust correctly in the more complex experiments.

Declaration of originality

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the School of Engineering at The University of Edinburgh.

Hugo Monteiro

Acknowledgements

During the period of this PhD many people have helped me one way or another. In these section I will name a few:

My supervisor Alan Murray, for the useful advice and motivation given throughout my PhD.

Dr. Bernd Porr and Dr. Les Haworth for accepting to be the examiners of this thesis and for the helpful comments that allowed increase its quality.

I would like to thank in particular to Andrew Cogman, Keith Muir and Alana Blewitt for helping me maintain some of my sanity during this period by providing very good company, usually accompanied by nice food and drinks. I would also like to thank Andrew Angus, Kenny (Cheng-Kai) Lu, Katherine Cameron and everybody else at the lab for all the fun conversations, support and for generally making the office a pleasant place to work in. I would like to thank all of them again for proof reading the several iterations of my thesis.

The Edinburgh University Rifle Club and its members, for enabling me to practise a very fulfilling and relaxing sport in very good company.

My family for their unconditional support throughout the whole period. And finally Sara, for giving me strength and for her never ending patience.

My PhD was funded by EPSRC and by Fundação para a Ciência e Tecnologia.

Contents

List of figures

Chapter 1
Introduction

Robots are becoming more popular every day, the number of applications increases at great speed, from vacuum cleaners to industrial robots in assembly lines, and surgical robots which can be used to operate remotely on a patient [1].

Autonomous moving robots can be particularly relevant for applications such as space exploration [2–4], hazardous environments [5], disaster zones [6], mainly applications where having a human present is not possible and/or inadvisable. These robots can be separated into several types according to the method of locomotion used. For ground robots, the most popular ones are wheeled, but there are also tracked robots, legged robots and other less used types.

Whilst wheeled and tracked robots are simple to control, legged robots are more flexible in the types of terrains they can handle. However, they are also more challenging to control than wheeled robots, which decreases the interest potential users have on them. Traditionally, legged robots have had very complex and computationally intensive control systems which, by taking precious power and computational time resources, puts a limit on how far they can be taken in an automation point of view.

As walking is the most basic task an automated legged robot has to accomplish before any other can be performed, finding better alternatives to the traditional control mechanisms becomes necessary in order to free up vital resources and create an opportunity to use more complex higher level automation tasks.

Nature has already solved the control problem for legged locomotion, so instead of trying to "reinvent the wheel", another approach can be attempted instead: to understand the principles in which that solution is based and use them to our advantage. The human brain is not as good at performing multiplications, matrix operations and other such functions performed easily by traditional computers which solve several million calculations per second however, one thing they accomplish very well

is interacting with the physical world and controlling the bodies in which they are integrated by using a large number of simple and low power computing nodes in a highly parallel architecture, the neurons. So it may be beneficial for the development of autonomous robots to take advantage of such an architecture. Neuromorphic systems attempt to mimic small components or processes of nervous systems from animals. These systems are usually amenable to analogue hardware implementation, which allows them to have very low energy consumption and high miniaturisation capability.

Another benefit of using biologically inspired mechanisms is the possibility of finding potential problems in the existing models, leading to their reformulation and offering additional insight on the quality of the current understanding of biology.

Some animals, such as the cockroach [7], rely on a mechanism called Central Pattern Generator (CPG) as the basic generator of the rhythmic movements involved in walking. Other animals, such as the stick insect [8–10] use a more distributed mechanism of control which relies more on proprioception to generate the movements and the synchronisation of the limbs. Both mechanisms have been trialled with success in a variety of robots [11, 12]. In animals, both of these types of mechanisms are then modulated by additional sensory signals in order to change their behaviour to suit the circumstances.

Animals also have reflex behaviours which are "hard coded" reactions to certain events. In locomotion there are a few reflexes regarding obstacles, such as steps or gaps, which come into action usually by being triggered by tactile sensors. These reflex behaviours are used as a basic modulation, but with time, animals start to recognise actions which lead to the situations causing the reflexes and learn how to adapt their actions in the presence of those predictive signals. One of these signals originates in the vision system and is very important for locomotion as, after learning, it allows the animal to react to problematic situations at a greater distance, before they require sudden reflexive actions to be taken.

The modulation of movements by sensory information such as vision is a very important component of locomotion in irregular terrain, for example with holes, steps, and other obstacles. It is also a very complex mechanism to hard code, as it depends on

the individual characteristics of the robot in which it is being implemented. The ideal system would adapt the modulation automatically on each individual robot by learning with experience without any need for intervention from the user. These types of learning systems are called Unsupervised Learning Systems.

It is in the context of this interaction between robotic platform, basic walking controller, learning mechanism and vision system which the work described on this thesis is performed.

1.1 Thesis statement

This project explores the suggestion that biologically inspired control and processing systems can improve hexapod walking behaviour on uneven surfaces by using a biologically inspired unsupervised learning algorithm to use sensor readings to adapt its gait.

1.2 Chapter layout

Chapter 2 gives an introduction to legged robots, their control types and sensor systems. Further background on the individual systems can be found at the beginning of each of the corresponding chapters.

Chapter 3 presents two different versions of the learning algorithm used for the terrain adaptation. The limitations of the first version are discussed and a second version which overcomes these limitations is proposed.

Chapter 4 starts by describing the "depth from motion" vision system which was originally intended to be used and mathematically explains the limitations which led to the decision of not using it in this context. It then describes the vision system and obstacle detection mechanism implemented as a replacement.

Chapter 5 explains the architecture used for the integration of all the systems, and gives the simulation results for several steps of the system integration.

In Chapter 6, the results are summarised and some conclusions are presented. Some potential areas for future work are also discussed.

Chapter 2
Robot Sensing and Control Review

As the work described in this thesis was developed in the context of a legged robot application using biologically inspired components, this chapter will give an introduction to several topics within this area, which will be of use for the understanding of the rest of this thesis.

To begin with, several types of locomotion will be discussed with their advantages and disadvantages, followed by a description of different biologically inspired control strategies for legged robots. Some sensing mechanisms will then be introduced.

2.1 Locomotion types

There are two main categories of land robot locomotion: wheeled and legged. Each has its own strengths and weaknesses [13, 14].

Wheeled robots are extremely efficient and easy to control when moving on smooth and flat surfaces, but have severe difficulties when the terrain is slightly more challenging, for example when its is rocky, has low coefficient of friction or has very steep inclines.

Legged robots are better suited for locomotion in difficult terrain, where the flexibility given by their legs allows them to choose better foot placements and maintain a levelled stance. On smooth and flat surfaces they have the disadvantage of being slower, less power efficient and more complex to control than wheeled ones. As legged robots have a higher number of degrees of freedom, they have greater flexibility in the types of movements they are able to make. They can for example go from walking in the ground to climbing a wall or a tree [15, 16]. These extra degrees of freedom can also be used to compensate a fault in one of the limbs [17–20].

This higher number of degrees of freedom available also has disadvantages, the main

one being the high complexity of the control system necessary to use them effectively. Developing such a system is a much more demanding task than developing the equivalent system in wheeled robots. They also require a higher number of moving parts and have a higher energy consumption (on terrains in which the wheeled vehicles can operate) [13].

Nevertheless, when the environment in which the robot is being used is uncertain, the potential benefits outweigh the difficulties. Legged robots have several potential applications. They can be very important in search and rescue scenarios within disaster zones [21], where the terrain is complicated, for example for searching for survivors inside the rubble of earthquake stricken buildings [22], thus improving accessibility and minimising the risk for the lives of the people involved in the rescue.

Another potential use is in space exploration. Using wheeled locomotion in space probes forces the adoption of more conservative navigation paths as it is more likely to encounter difficulties in problematic areas like rocky terrain and sand areas. These problems would be reduced if legged locomotion was a possibility. In fact, there is already a NASA JPL project which is developing a hybrid robot called ATHLETE (All-Terrain Hex-Limbed Extra-Terrestrial Explorer). This robot has six legs and small wheels at the end of each leg, so it can be efficient on smooth surfaces, but still be able to overcome difficult terrain [23].

As a requirement of this project was biological plausibility, and since there are no animals using wheels as a form of locomotion, the next part of this chapter will describe several types of legged locomotion as well as types of control for legged locomotion.

2.1.1 Legged Robots

Depending on the number of legs a robot has, it can perform different types of gaits (movement patterns of the different legs), which give it different speed and stability. The different characteristics of the most common types of robots (with 2, 4 and 6 legs) are described below.

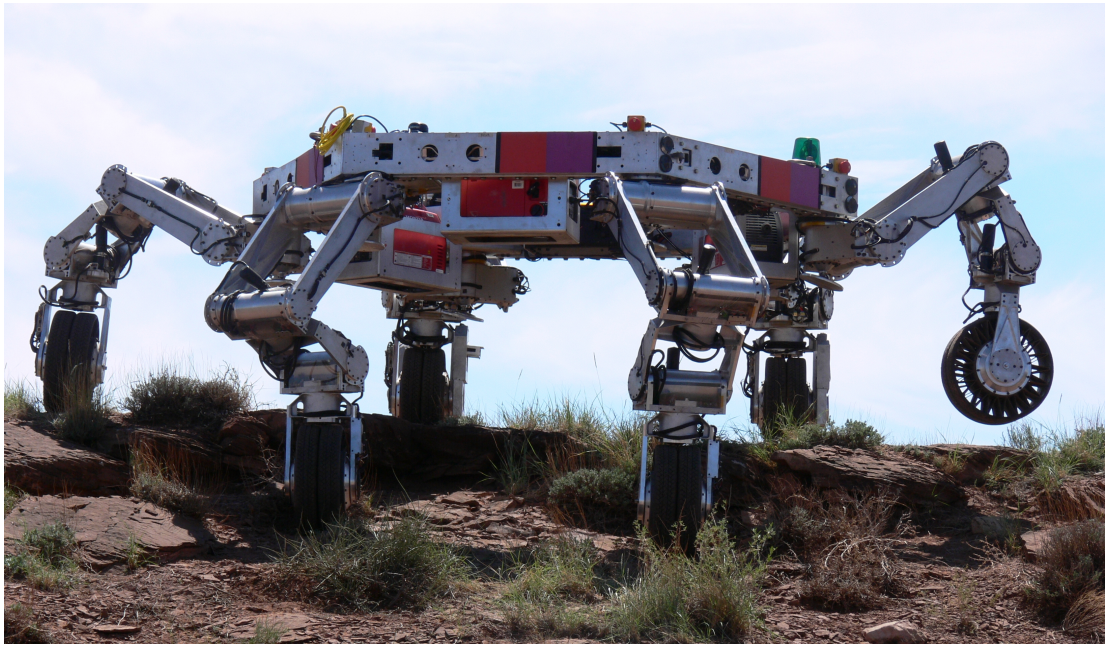


Figure 2.1: ATHLETE robot prototype *(picture from [24]) developed by NASA-JPL.*

2.1.1.1 Biped

Biped locomotion, with its limited number of support points is mostly based on dynamic stability, which means that, even when standing, balance is achieved by constantly adjusting muscle/motor tension to compensate for any deviation from the vertical position. A very famous example of a biped robot is ASIMO from Honda, which can be seen in figure 2.2

The main types of biped locomotion are walking and running. When walking (figure 2.3a), the robot alternates between one and two support points. At the initial position, potential energy is converted to kinetic energy up to the point where the other leg reaches the ground. At this point, the second leg starts supporting the body and the kinetic energy is converted to potential energy again. The body then starts to rise until it reaches to the vertical position again, when the cycle restarts again. This cycle is similar to an inverted pendulum [26].

When running (figure 2.3b), the gait changes to one where moments of single support alternate with moments of suspension, when no part of the body is in contact with the ground. In this gait type, the leg is analogous to a spring, accumulating the kinetic energy into elastic energy and then releasing it again in kinetic energy form [26].

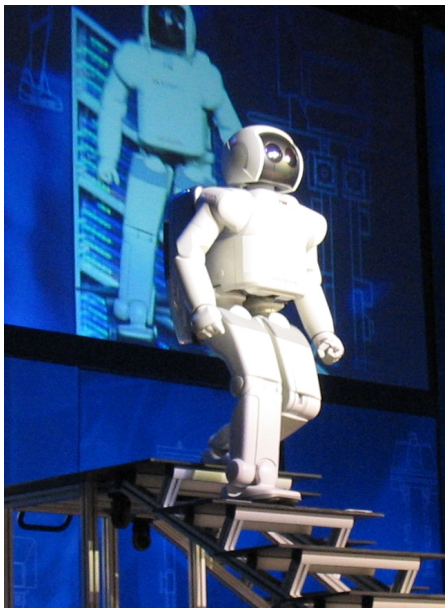


Figure 2.2: ASIMO robot *(picture from [25]) developed by Honda.*

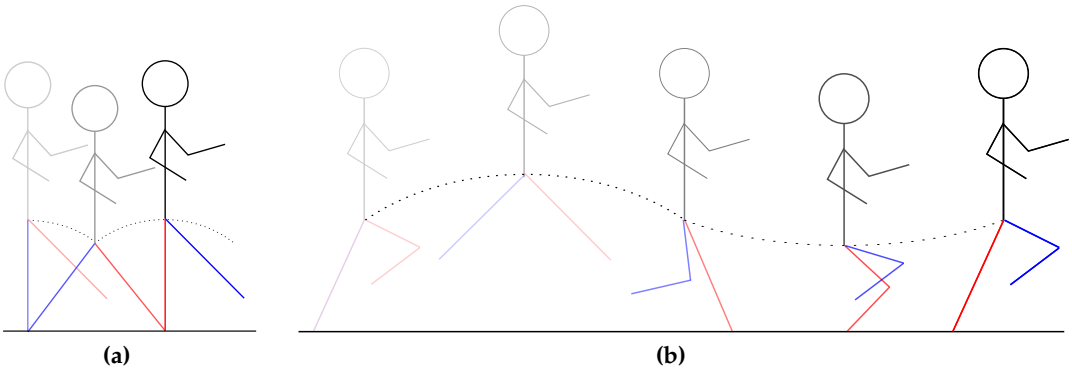


Figure 2.3: Biped walking (a) and running (b) gaits.

2.1.1.2 Quadruped

In robots with four or more legs, it becomes possible to have walking patterns based upon static stability [27,28]. Static stability occurs when the centre of gravity's vertical projection lies inside the base of support, which consists in the convex hull formed by the points of contact of the feet with the ground.

In quadruped robots, taking advantage of this type of stability means the robot would only be able to move one leg at a time (creeping gate), thus making the progress through the terrain very slow [27, 28]. Because of this limitation, walking patterns based on dynamic stability are still preferred.

Recently, a quadruped robot which has generated a lot of interest from the general public is a military prototype called Big Dog [29–31], developed by Boston Dynamics (figure 2.4). This robot is able to walk in challenging terrain, like snow and ice, and is able to recover from external perturbations like a side impact.

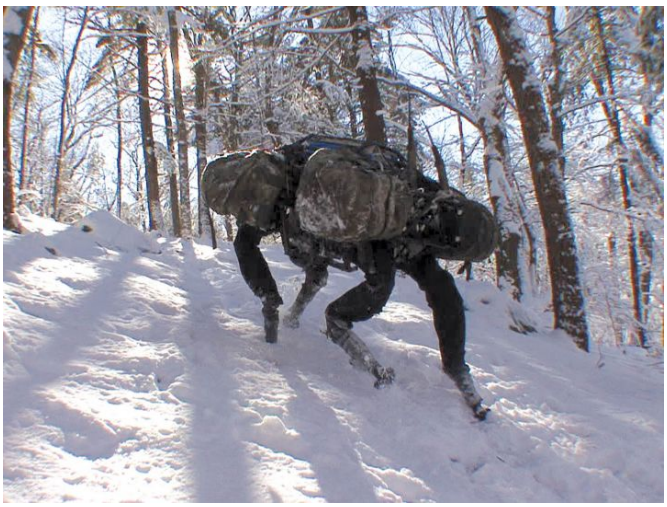


Figure 2.4: Big Dog robot *(picture from [32]) developed by Boston Dynamics for the United States Marine Corps.*

2.1.1.3 Hexapod

Hexapod robots are based upon the structure of insects. Hexapod robots, or, in fact, any robot with six or more legs, are able to use static stability while walking efficiently and fast. This happens due to the possibility of forming several independent triangles with the supporting legs which contain the centre of gravity's vertical projection [28].

The biggest motivation for the use of hexapodal robots is therefore the ability to implement a reasonable walking system without having to focus on solving problems such as balance. This saved effort can instead be used in the attempt to improve the systems performance on a variety of environments. The extra legs also offer an advantage when the loss of a limb occurs, in which case, the robot still has the ability to continue using a less effective, but still functional and stable gait [18,20,33].

All the reasons above make the hexapod platform a popular choice amongst the research community and some examples are presented in figure 2.5.

(a) Hector [34] **(b)** Tarry II [35] **(c)** Robot V [36]

Figure 2.5: Examples of existing hexapod robots.

Some of the most used gaits in robotic hexapodal locomotion are the wave, ripple and alternating tripod gaits [37].

The fastest gait is the alternating tripod, followed by the ripple gate and the slowest among these is the wave gait. In terms of stability, it follows the reverse trend, with wave gait being the most stable, as it provides more points of support, and the tripod gait on the other end of the scale as the least stable, as it uses only the minimum 3 points of support in all but short occasions [37].

The **Wave gait** is the simplest and consists of moving only one leg at a time.

The **Ripple gate** consists in moving 2 legs at the same time, one on each side. As this movement spreads through the rest of the legs on each side, it looks like a ripple propagating.

Alternating tripod gate is the fastest gate. With six legs it becomes possible to form two large independent triangles, one formed by the front and rear left legs and the middle right leg, and the other by the front and rear right legs and the middle left leg.

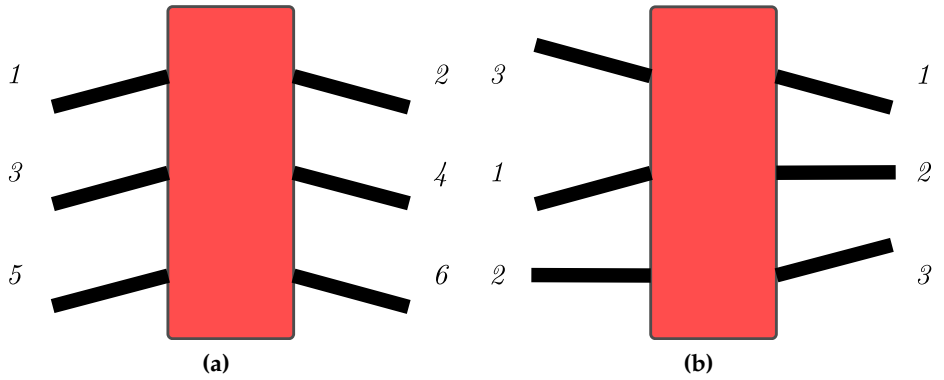


Figure 2.6: Wave (a) and Ripple (b) gaits

When one group is sustaining the body weight and moving back, therefore moving the body forwards, the other group is lifted and moving forward. This allows for the body to be always moving while maintaining a steady position.

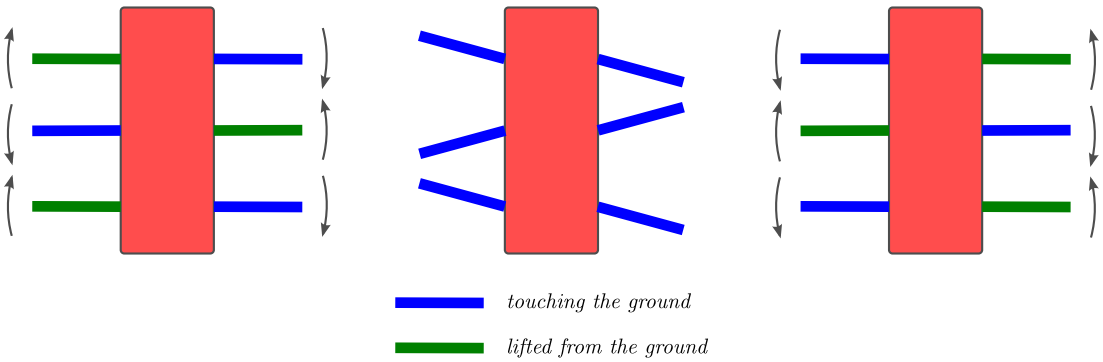


Figure 2.7: Alternating tripod gait. *It consists in alternating between the first and last of the statuses represented. The legs represented in blue are supporting the body weight while the ones in green are lifted.*

Sensory feedback systems allow for a more dynamic locomotion that changes with the circumstances, of which, Sensory Coupled Action Switching Modules [11,38] is an example which uses the equivalent of state machines whose transitions are controlled by sensory input, like joint position or load. They tend to converge on a tripod gait as they speed up through a flat terrain, whilst returning to slower and more stable gaits when faced with more difficult terrain, like slopes.

2.2 Control Types

There are several options available for the control of a legged robot. On one end are the systems which plan each foot placement to be taken to the level of the individual angles of each joint, on the other are the systems which rely on simple biologically inspired mechanisms which need very little computation.

The complex analytical systems have the advantage of being more accurate in their actions, at the expense of the need of very high processing power, and since the resources are usually limited, usually there is not much room for other complex high level control systems. This type of control is adequate for semi-autonomous robots, which rely on external systems to control higher level functions.

The biologically inspired systems take advantage of simple mechanisms and rules to control the robot locomotion, which require less processing power, therefore releasing more resources to other systems. This characteristic gives them the potential to use more complex high level systems or to increase their power autonomy.

In hexapod robots, the main inspiration for leg control comes from insects, which have a varying mixture of two different systems, Central Pattern Generators (CPGs) [39,40] and Sensory Feedback systems [41].

CPGs are mechanisms which generate gaited walking from the rhythm produced by a central system ([39] and [40] provide a comprehensive review of the use of CPGs in animals and robots), whilst Sensory Feedback systems are more of a distributed system, in which each joint has its own control system which operates based on its local sensory information [33, 41, 42]. These tend to converge to a gaited walking on smooth surfaces and diverge to other adaptive walking patterns when faced with rough terrain.

Walknet [33, 42] is an example of a distributed control system which uses artificial neural networks to emulate the behaviour of a stick insect. This system falls in the Sensory feedback system category, as the movements are generated without the use of an external oscillatory signal and are instead generated through the processing of the system status and sensor readings in an artificial neural network that calculates the change in position.

Both CPGs and Sensory Feedback System have the possibility of being modulated by external signals in order to change the gait characteristics in response to stimuli, for example when approaching an obstacle. As these external stimuli can be of a wide variety, it is not possible to program all the behaviours beforehand, but it is possible to incorporate both simple reflex behaviours, which happen for example when the robot bumps into the obstacle, and have a separate mechanism to learn how to improve the gait modulations in order to avoid the collisions in the first place. This topic will be further discussed in Chapter 3.

2.3 Sensing

Most robots have sensory mechanisms in order to acquire information about the surrounding environment, the most common ones are tactile and vision sensors. The tactile sensors are usually pressure triggers or torque sensors on the joint motors, but robots with antennas serving as touch sensors can also be found [43]. The vision sensors are usually single or dual camera setups. Chapter 4 explains in more detail on how the vision system can be used in robots.

One of the systems tested in this thesis (section 4.1) was a commercially available artificial neuromorphic retina [44, 45]. Artificial neuromorphic retinas are usually based on detecting temporal or spatial changes, with some implementations [44, 46, 47], using asynchronous events rather than frames. This asynchronous behaviour has two advantages: (1) it reduces the average bandwidth necessary to transmit images by only transmitting the data corresponding to the pixels that have changed, and (2) it is able to detect very fast movements, as any change comes through at the time it is detected and does not have to wait for a clock to activate, as it happens in frame based cameras.

Chapter 3
Learning

"Learning, the alteration of behaviour as a result of individual experience."
in *(ENCYCLOPÆDIA Britannica)*

In this chapter, several types of Reinforcement Learning and Unsupervised learning will be described.

Reinforcement learning affects the decision making process by taking into account past experience to change how future actions are taken. It relies on evaluative feedback (either via a punishment or a reward which can be internal processes such as pain or pleasure or external ones) for its actions in order to make judgements on how effective its action was. The agent will then use this information to adjust its behaviour accordingly in future actions. [48]. This allows it to be used even when the agent cannot be taught beforehand, as it adapts itself with experience.

Unsupervised learning consists of learning pure correlations in data or signals. What sets it apart from the other types of learning is that it does not rely on a goal or any sort of feedback. Instead, it is mostly based on internal processes which detect correlations between events to infer causality and update the behaviour. Pure Hebbian learning [49] is an example of such type of algorithm.

The type of learning which hasn't been mentioned yet is Supervised learning, which consists in learning from examples already classified as good or bad. It is an important type of learning, yet it is not sufficient to reach autonomy in the environment as there are always unforeseen situations which cannot be generalised from the training data and which the agent will have to learn by trial and error (Reinforcement Learning). It also suffers from not having a clear parallelism to biological mechanisms [50].

In the last two decades, many of the theoretical models used in reinforcement learning

have either been proved to take place in the physical systems or have laboratory analysis of the physical systems [51,52].

This work was undertaken as part of a project whose main focus was to use these biologically inspired learning mechanisms in conjunction with other biologically inspired mechanisms, such as a biologically inspired robotic platform, vision and tactile systems.

In animals, the basis for learning lies in the change of connection strength between synapses in the brain. This mechanism is called synaptic plasticity [52]. Some examples of different types of learning involving synaptic plasticity will be explained in the next few sections of this chapter.

3.1 Hebbian Learning

Hebbian Learning is an unsupervised learning mechanism based on correlation detection. It is also the basis for all the other mechanisms described in this chapter.

This type of learning can be described by the following quote from its creator, Donald Olding Hebb [49]:

> "When an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased".

This rule is usually translated into mathematical form by using equations similar to 3.1, 3.2.

Considering there are N presynaptic neurons, whose outputs are represented by u_i where $i = 0 \ldots N$ and the output for the postsynaptic neuron is represented by v, the association between these parameters is given by the following equation:

$$v = \sum_{i=0}^{N} \rho_i u_i \tag{3.1}$$

Where ρ_i are the weights modulating the synaptic connections between pre and postsynaptic neurons.

In this project, following on the strategy used in [53,54], the reflex signal is considered a special input, in the way that it is considered to be a static behaviour that does not change and the goal of the learning system is to associate other sensors with these reflexes, predict the situations in which they arise and avoid them by anticipating an action (figure 3.1). Therefore, in the remaining of this chapter, the presynaptic signals are distinguished between two different sets of pathways, a reflex pathway, represented with index 0 and the predictive pathways, with index between 1 and N. For this work, the reflexive pathway is considered to always elicit the same response in the postsynaptic neuron, so its weight is fixed and only the predictive pathways are subjected to learning. This small change makes the system leave the realm of unsupervised learning and into machine learning, as in this situation, the reflex is seen as a punishment, which should be avoided.

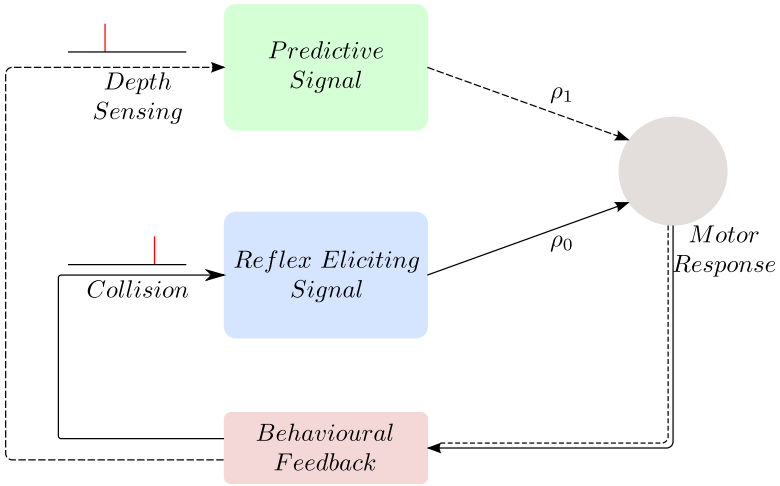


Figure 3.1: Learning Architecture *(adapted from [55]) with 1 reflex and 1 predictive pathways. The reflex loop encodes a reactive action to the collision whose timing association with the predictive loop will be learnt by the system in order to avoid the reflex events.*

Considering these constraints, in Hebbian learning, the weight is then updated according to the amount of simultaneous activity between pre and postsynaptic neurons, like so:

$$\frac{d\rho_i}{dt} = \mu u_i v, \quad i = 1 \dots N \tag{3.2}$$

Where μ represents the learning rate which is used to control how fast the learning process takes place, (too fast and it will overshoot the optimum level, too slow and it will require an excessive amount of trials in order to achieve learning).

Replacing equation 3.1 in 3.2 the following equation representing the weight update rule is obtained:

$$\frac{d\rho_i}{dt} = \sum_{j=0}^{N} \mu u_i \rho_j u_j, \quad i = 1 \ldots N \tag{3.3}$$

If a simple case with only one reflex pathway, together with one predictive pathway is considered, it can be represented diagrammatically by figure 3.2.

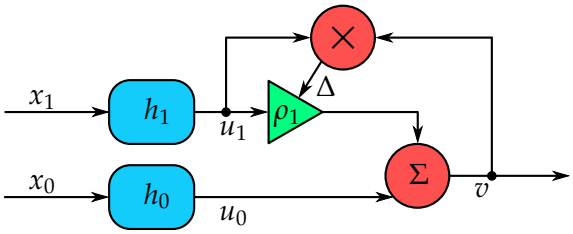


Figure 3.2: Architecture of Hebbian Learning *with 1 reflex and 1 predictive pathways. x_0 and x_1 represent the reflex input and predictive input respectively. h_0 and h_1 are the bandpass filter which create traces u_0 and u_1 of the input spikes. ρ_1 is the weight of the predictive pathway, which is being updated by the learning algorithm. Σ represents the neuron, which integrates the input signals and v is its output.*

3.1.1 Stability

Hebbian learning is known to be unstable due to the positive feedback process that occurs, where higher weights lead to higher correlation, which further increases the weights. This can be better explained by expanding equation 3.3 for the case where there is 1 predictive input and 1 reflex input.

$$\frac{d\rho_1}{dt} = \mu u_1 \rho_0 u_0 + \mu u_1 \rho_1 u_1, \tag{3.4}$$

As can be seen by the second term of the equation ($\mu u_1 \rho_1 u_1$), as soon as the predictive weight (ρ_1) becomes non-zero, it no longer needs a reflex to continue growing, as the signal will correlate with itself.

Many attempts have been made at stabilising this form of learning (e.g. [56–59]).

The Oja rule [56] and the BCM rule [57] (both explained in further detail below) are examples of important contributions towards the stability of Hebbian Learning, which inspired many others. Other types of attempts are weight decay, weight limitation and applying resource constraints [50, 60, 61].

Oja rule

The Oja rule is a mechanism used to stabilise Hebbian learning. In [56] it was proposed by Oja that the weight increase should be balanced by a depressive force proportional to the postsynaptic activity. This successfully decreases the implicit exponential weight growth present in common Hebbian Learning.

$$\frac{d\rho_i}{dt} = \mu \left(u_i v - v^2 \rho_i \right) \tag{3.5}$$

This specific learning rule has also an interesting effect of performing PCA (principal component analysis) over the inputs. More recently, this rule was shown to be partially analogous to a biological mechanism of synaptic scaling [52].

BCM rule

The BCM rule [57] (whose name also derives from the names of its creators) uses a dynamic threshold (θ_M) in order to decide to increase or decrease synaptic weights depending on whether the postsynaptic activity is above or below this threshold respectively. The threshold changes depending on the sliding average of the postsynaptic activity (E). When the postsynaptic activity is high, the threshold increases, thus increasing the probability of the weight decrease occurring and when the activity decreases, the threshold lowers, thus increasing the likelihood of an increase in the weights.

$$\begin{aligned} \frac{d\rho_i}{dt} &= (v - \theta_M)\, u_i v - \epsilon \rho_i \\ \theta_M &= E\left[(v/v_o)\right] \end{aligned} \tag{3.6}$$

3.2 Differential Hebbian Learning

While traditional Hebbian Learning uses the correlation between the inputs and the output to change the synapse weights, which produces a symmetrical weight curve with regards to the order of the events, there is a derived algorithm called Differential Hebbian learning which allows for an asymmetrical learning. The **Differential Hebbian** learning was first created by Bart Kosko in [62] (shortly after this, another system which also took advantage of an asymmetric learning rule is [63]). This type of learning uses the correlation between the inputs and the amount of change they produce in the output. This type of mechanism has been found to be connected to another biological mechanism, Spike-timing-dependent plasticity (STDP) which will be described in section 3.3.

The same notation and restrictions used to describe Hebbian learning are applied here with the addition of the output variation which will be represented from now on by v'. This output variation is explained as a function of the inputs by the following equation:

$$v' = \sum_{j=0}^{N} \rho_j u'_j \tag{3.7}$$

Where u'_j represents the derivative of input u_j. The weight variation which leads to the learning can be represented by the equation below:

$$\frac{d\rho_i}{dt} = \mu u_i v', \quad i = 1 \dots N \tag{3.8}$$

If equation 3.7 is substituted in equation 3.8, the following is obtained:

$$\frac{d\rho_j}{dt} = \sum_{j=0}^{N} \mu u_i \rho_j u'_j, \quad i = 1 \dots N \tag{3.9}$$

This law has inspired a new family of rules, among those, ISO Learning [54, 55], ICO Learning [64, 65] and ISO3 Learning [66–68], which have been developed by Bernd Porr and Florentin Wörgötter (one of the project collaborators).

Isotropic Sequence Order (ISO) Learning

ISO Learning [54, 55] essentially implements the Differential Hebbian Rule, with small, but important, changes. In ISO learning, both the predictive ($x_i,\ i = 1 \dots N$) and reflexive (x_0) inputs are spikes instead of firing rates and these are passed through band-pass filters (h) in order to create traces u_1 and u_0 respectively.

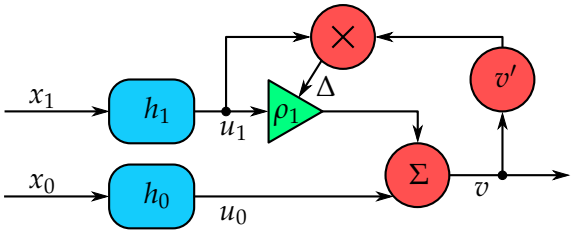


Figure 3.3: Architecture of ISO Learning *with 1 reflex (x_0) and 1 predictive (x_1) pathways. The difference to Hebbian learning, described previously in the section, lies in the fact that in this case the weight update depends on the amount of variation of the output (v') instead of its absolute value.*

When these filters obey certain rules, the auto-correlation component, present in Hebbian learning and also in its Differential version, disappears as it has been demonstrated in [68] due to the integral of the filtered signal being 0 which means that without correlation, the signals will have no impact on the weights. Yet, it is still very sensitive to small errors, as these will cause the integral to be non-zero. Numerical approximations or small noise levels will cause it to have an exponential component in the weight growth as seen in the results from the original articles [54, 55]. Figure 3.3 illustrates this algorithm.

ICO Learning

ICO Learning was developed as an attempt to solve the auto-correlation problem still present in ISO learning due to its numerical instability. This was achieved by using only input correlations in the calculation of the weight change [64, 65].

The weight update for ICO Learning can be described by:

$$\frac{d\rho_i}{dt} = \mu u_i u_0', \quad i = 1 \dots N \tag{3.10}$$

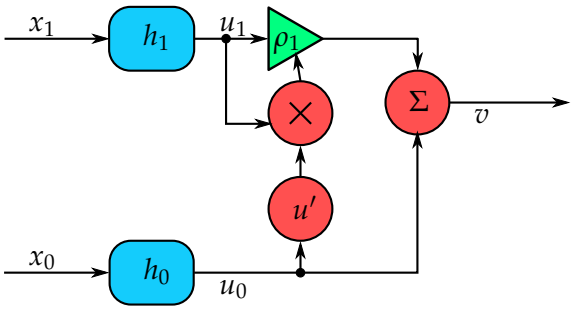


Figure 3.4: Architecture of ICO Learning. *The difference from this algorithm to the ISO learning is in that, instead of using the output variation of the neuron to modulate the weight change, it uses the variation of the reflex signal directly.*

Although this algorithm no longer suffers from instability problems, it comes at the expense of biological plausibility, as explained by Kolodziejski et al. in [68] where it is mentioned that "only a few specialised synapses" implement this type of learning. An illustration of the algorithm's architecture can be seen in figure 3.4.

Third Factor ISO Learning (ISO3)

ISO3 [66–68] solves the auto-correlation problem by taking a different approach. Instead of changing the architecture of the algorithm significantly, ISO3 adds an extra component to standard ISO in order to take advantage of certain properties of this learning mechanism. Its architecture is illustrated in figure 3.5.

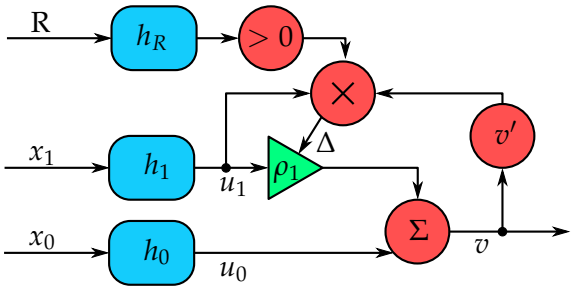


Figure 3.5: Architecture of ISO3 Learning. *The main difference from this algorithm to the ISO Learning is the addition of a third input, called relevance, which controls the time at which the learning takes place. By controlling the timing of the learning, the auto-correlation can be minimised.*

As can be seen in figure 3.6, the bulk of the auto and cross correlations happen at different moments in time, the auto-correlation happens mostly between the prediction spike and the reflex spike and the cross correlation happens mostly immediately

after the reflex spike.

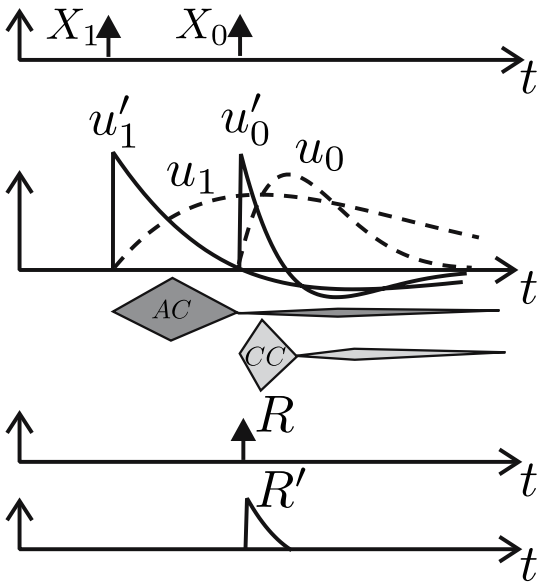


Figure 3.6: ISO3 Timing *(figure adapted from [68]) - the auto-correlation component (AC) in ISO learning happens mostly between the prediction spike (X_1) and the reflex spike (X_0) and the cross-correlation (CC) is stronger in a period right after the reflex spike. This allows the third factor (R), introduced by ISO3, to limit the learning to the period of strongest cross-correlation and weakest auto-correlation.*

ISO3 takes advantage of this property by inserting a third type of signal R (for "Relevance") whose function is to limit the learning to the period during which the cross-correlation is maximum and the auto-correlation is almost non-existent. An analogy to biological systems can be made, as it is known that certain chemical signals control the timing at which learning takes place [66–68].

The weight update for ISO3 is represented by equation 3.11.

$$\frac{d\rho_i}{dt} = \mu u_i v' R' \theta\left(R'\right), \quad i = 1 \ldots N \tag{3.11}$$

In practise, the signal used as a third factor is in fact the reflex signal, which is used in this role merely as a timing signal to turn on learning. This algorithm has been shown to work in experimental implementations where it was used to anticipate simple reflex behaviours in wheeled robots [64, 66, 67, 69]

3.3 Spike Timing Dependent Plasticity

Another learning system worth mentioning is Spike Timing Dependent Plasticity (STDP) [70]. This is based on a biological process with the same name, which takes place at a neuron level. It detects causality and changes the synaptic weights accordingly. It has an asymmetric learning rule represented by figure 3.7. According to whether the spike arrives before or after the postsynaptic one, it can cause one of two effects: long term potentiation or long term depression.

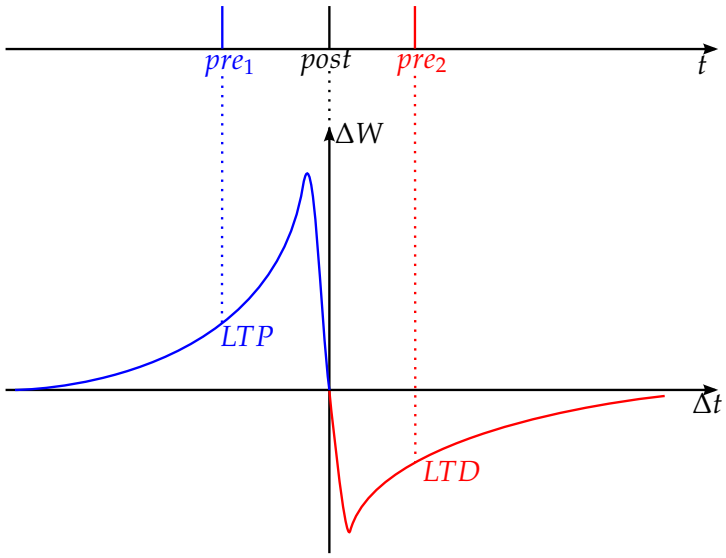


Figure 3.7: STDP example curve *- If the presynaptic spike fires before the postsynaptic spike (pre_1), the synapse will suffer Long Term Potentiation (LTP), otherwise (pre_2) Long Term Depression (LTD) will occur.*

Long term potentiation (LTP) takes place when the presynaptic spike comes a few milliseconds before the postsynaptic spike, causing the synaptic weight to be increased.

Long term depression (LTD) is the opposite of LTP. In this case, when the presynaptic spike comes a few milliseconds after the postsynaptic spike, the synaptic weight is reduced.

The exact timing constraints vary across the brain areas and can go from only a few milliseconds to hundreds of milliseconds [71].

3.4 Learning System Implementation

3.4.1 Motivation

As explained in section 3.2, ISO3 manages to improve on learning stability while maintaining biological plausibility. It has an inconsistency though, this algorithm accepts spikes as inputs, but then makes use of a linear model for the postsynaptic neuron. In order to improve the consistency of the model and make it more biologically plausible, it was replaced, in this project, with a spiking model. The spiking model chosen was of the integrate and fire type. The integrate and fire neuron model is one of the simplest spiking models available. It integrates the synaptic input values in the membrane until it reaches a predefined threshold, at which point it fires a spike (more information on neuron models can be found in [72] and [73]).

This system was then tested to verify whether this change maintained the same characteristics of the original learning rule.

3.4.2 Implementation of the Spiking ISO3

As already mentioned, the neuron model used in the original method was replaced by a very simple integrate and fire neuron model. The connections to this neuron were tuned in a manner so one spike from a presynaptic neuron would elicit multiple spikes at the postsynaptic output, the frequency of these spikes is proportional to the input level of the trace. This was performed in order to help preserve the characteristics of the model, which assumes a linearity in the relation between the input levels and the output activity.

For these tests, the following parameters were used:

- Predictive signal band pass filter: $f = 0.003$ and $Q = 0.501$
- Reflex signal band pass filter: $f = 0.008$ and $Q = 0.501$
- The learning rate was set to be $\mu = 0.015$
- The neuron was set to fire when the integration of the input traces reached 1.0
- All band pass filters were normalised to have a peak of value 1.0

In order to test the model, an open loop simulation, analogous to the one used in [67], was performed. For this simulation, a sequence of correlated spike pairs (figures 3.9 and 3.8) was used, followed by a period where the reflex action was removed to symbolise successful learning.

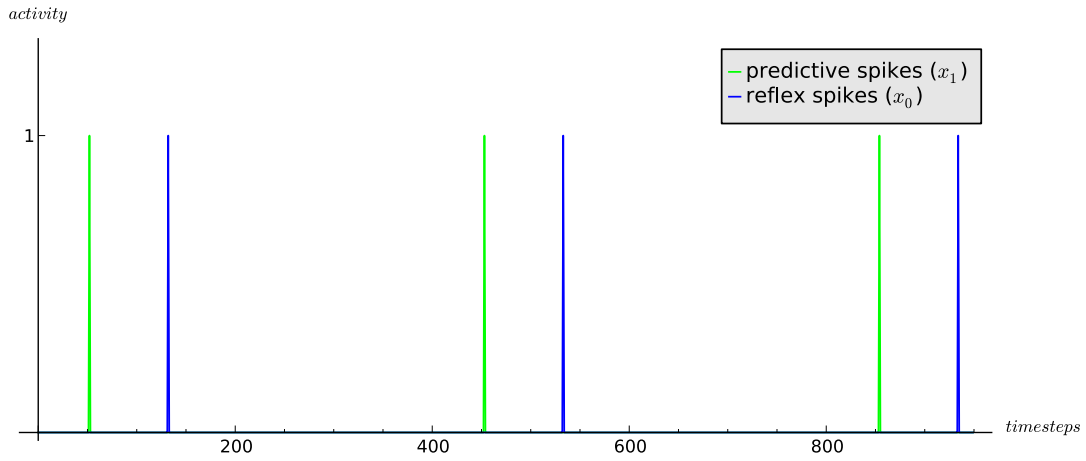


Figure 3.8: Correlation Pairs. *Detail of the pairs of spikes used for the open loop simulation of the spiking version of the ISO3 learning. The predictive spikes are shown in green and the reflex spikes are shown in blue.*

Figure 3.10 shows the weight growth during the simulation, as well as the stability of the weight after the reflex input is removed. It can be seen that the weight growth is quite similar to the one occurring in the original implementation [67] and that the stability properties are maintained as well. The detail of the correlation between the traces of the predictive spike and reflex spike can be observed in figure 3.11.

After confirming the learning system behaved as expected, it was integrated into the final control system. More details on this implementation and corresponding results are in Chapter 5.

3.4.3 Problems observed

After integrating the learning system with the robot simulations, it was observed that in certain scenarios, a problem would manifest.

This problem was detected in situations where the robot was trained with obstacles which regularly stimulated several of the predictive pathways immediately before

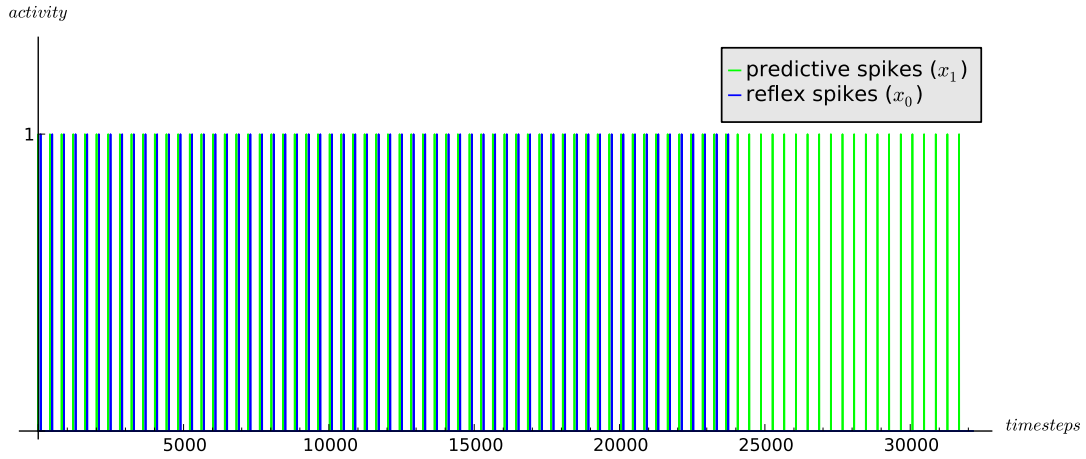


Figure 3.9: Input Spikes. *Spike train used for the open loop simulation of the spiking version of the ISO3 learning. It is composed of a series of the spikes pairs shown in figure 3.8 followed by a shorter sequence of singles spikes from the predictive pathway. As in the detail shown previously, the predictive spikes are shown in green and the reflex spikes in blue.*

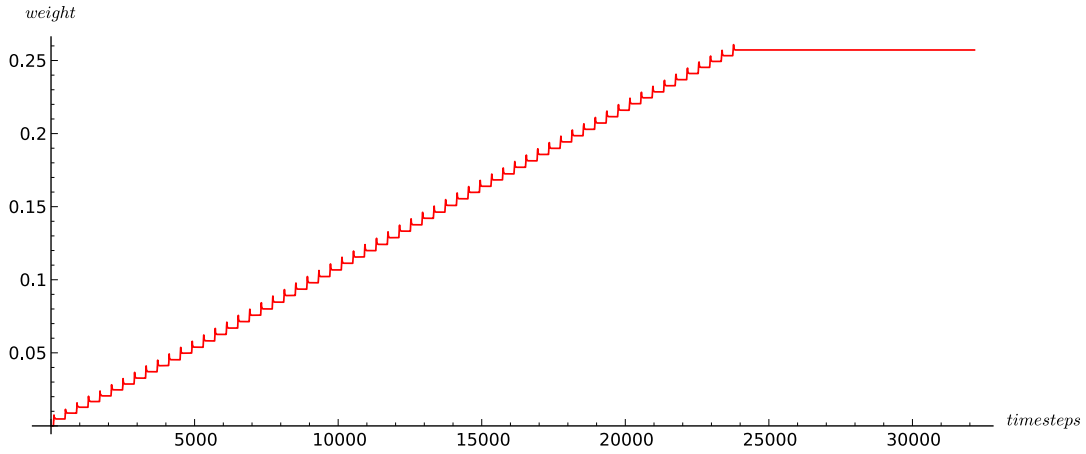


Figure 3.10: Weight progression *during the open loop simulation of the spiking version of the ISO3 learning*

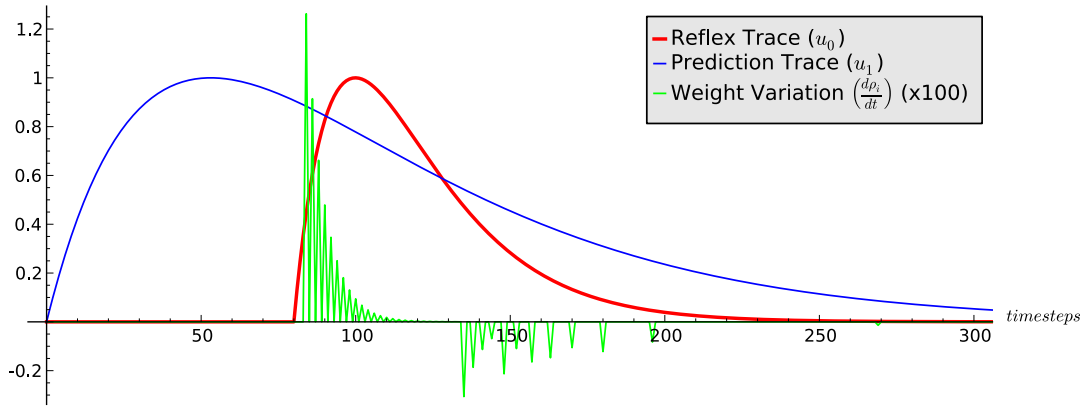


Figure 3.11: Weight update for the spiking version of the ISO3 learning. *Here, it is shown the shape of the weight variation* $\left(\frac{d\rho_i}{dt}\right)$ *for the input traces represented* (u_0 *and* u_1)

the reflex action occurred. Some of these inputs did not have a causal or predictive characteristic, as they were merely coincidental, but they were correlated to the reflex action by the system, which produced a weight increase on those connections.

This behaviour is expected, as at that moment the system does not have any information which would indicate these coincidental inputs were not predicting the activity.

The problematic behaviour is the lack of adjustment once the coincidental inputs no longer correlate with the activity. Due to this fact, when training proceeded to other types of situations where these same pathways were activated due to other events, not relevant to the activity of the robot, it would elicit a wrong reaction on its part. And independently of the number of runs in the training, this erroneous behaviour would not disappear.

This problem occurs due to the way learning takes place in ISO3. As explained before, learning only takes place when a reflex action also takes place, so, when other events happen to coincide with the reflex actions, their synaptic connections will also get strengthened. Since ISO3 does not have any mechanism to allow natural weight decrease, thus not being able to forget correlations, or any way to change the weights without the occurrence of a reflex action, the weights of pathways which get reinforced by coincidence and not actual causal relationships will tend to be increased, which causes the appearance of incorrect behaviours.

In order to illustrate this problem in more detail, a few open loop simulations were performed using artificial inputs and the ISO3 learning block.

For these tests, the following parameters were used:

- Reflex signal (x_0) band pass filter: $f = 0.008$ and $Q = 0.501$
- Predictive signal (x_1) band pass filter: $f = 0.003$ and $Q = 0.501$
- Erroneous signal (x_2) band pass filter: $f = 0.003$ and $Q = 0.501$
- The learning rate was set to be $\mu = 0.2$
- The neuron was set to fire when the integration of the input traces reached 2.5
- All band pass filters were normalised to have a peak of value 1.0

In the following simulations, the input pathways used are the usual reflex; a predictive pathway, which carries spikes that predict the reflex action; and a new error pathway, which carries spikes unrelated to the reflex action(figure 3.12).

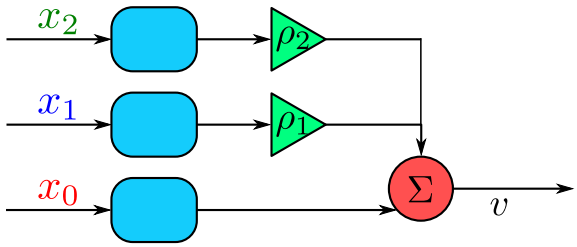


Figure 3.12: Inputs used during the open loop simulations used for debugging. *Besides the normal reflex (x_0) and predictive (x_1) pathways, a new erroneous (x_2) pathway is used. This pathway will be used to insert into the system spikes which are independent from the effect and which any correlation existence is not causal, but pure coincidence*

In the first simulation, the spike trains used represented the situation in which the problem was detected. To begin with, the spike triplets shown in figure 3.13 were presented to the system for several iterations. The error spikes were then removed for the rest of the learning (the inputs return to the format illustrated in figure 3.8).

After some iterations the reflex spikes are also removed and the effects of both the predictive and error pathways are tested individually. The full spike train used in the simulation can be seen in figure 3.14.

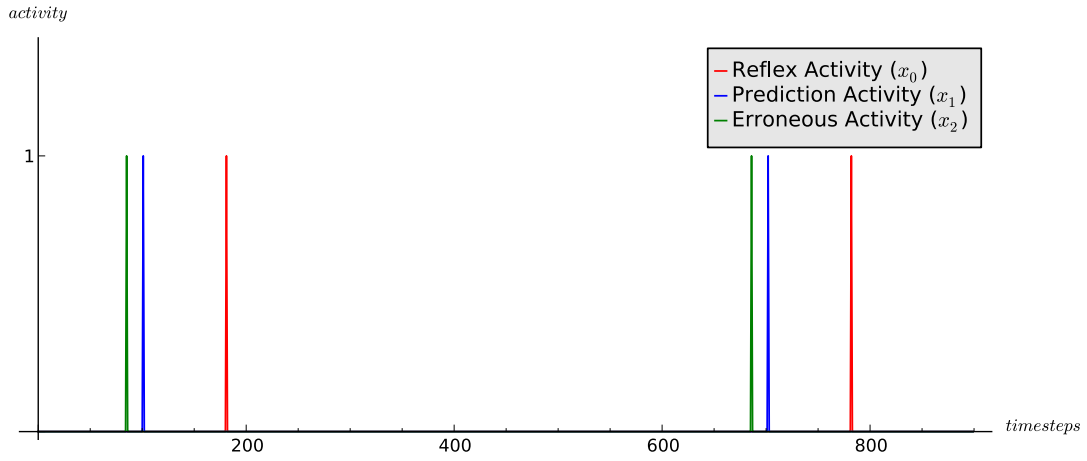


Figure 3.13: Correlation triplets. *The inputs for the initial iterations are now triplets formed by an extra coincidence input added to the already existing prediction and reflex inputs*

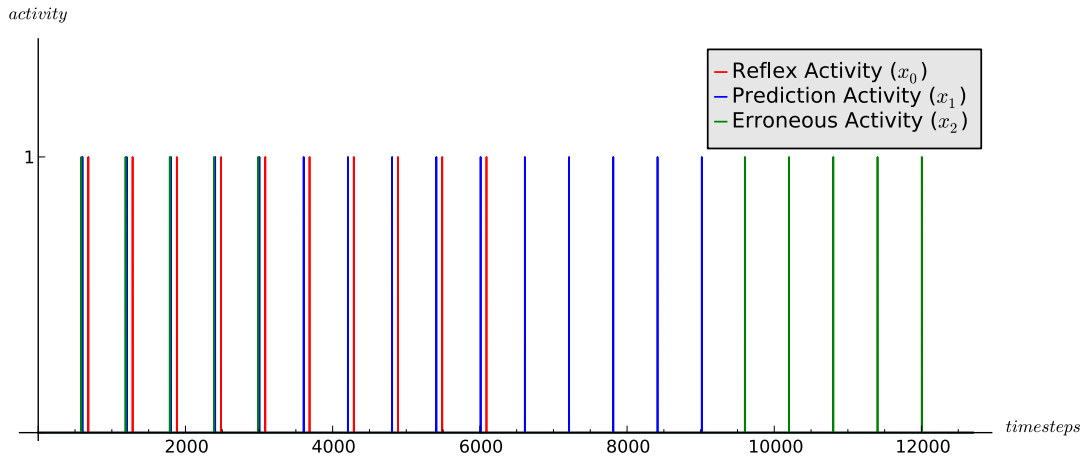


Figure 3.14: Inputs for the learning with coincidental erroneous inputs. *The input train is made of 5 triplets (show in more detail in figure 3.13) followed by 5 pairs illustrated in figure 3.8 followed by 5 individual spikes in the prediction input, followed by 5 individual spikes in the extra input*

As can be seen in figure 3.15, when the erroneous inputs are used after learning, they produce activity from the system due to the few correlations occurring early in the learning phase. These correlations increased the weights of this input path, which then maintained their level (figure 3.16) due to ISO3 learning not possessing any mechanism for weight decrease.

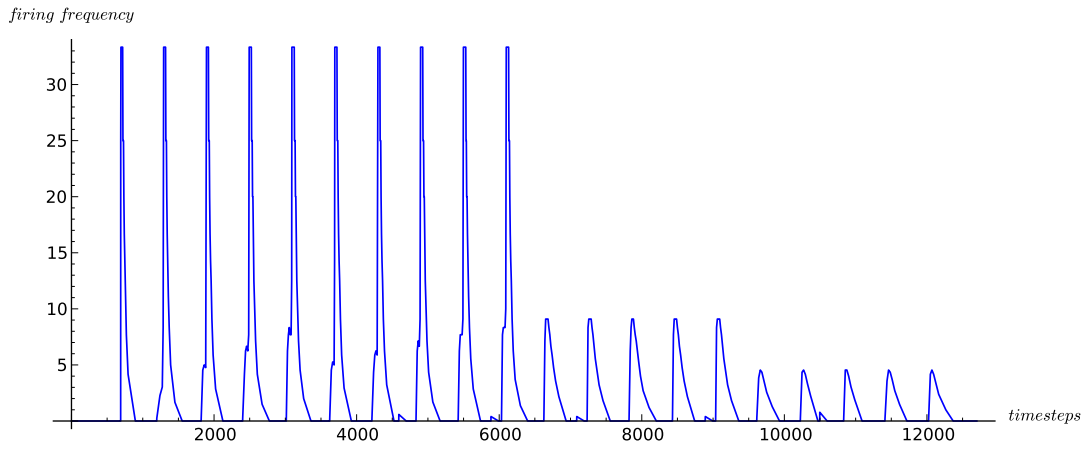


Figure 3.15: Output activity for Spiking ISO3. *Frequency plot of the ISO3 neuron output obtained when the spike train in figure 3.14 is used. It can be seen that there is some activity present when the erroneous spikes are presented after learning.*

3.4.4 Solution proposed - Forgetful ISO3

The main cause of the problem was identified as the lack of learning mechanisms capable of managing inputs that correlate with the reflex action, not by causality, but by chance and which limit the growth of the corresponding weights.

In order to illustrate this problem a simple example can be used. If a person approaches a hot plate with its hand, she will feel an increase in heat as it gets closer and when the hand touches the plate, the pain will cause a reflex to occur and the sensation of increasing heat will be associated with the pain of touching the source of the heat. If right before the touch a light flashes, this flashing light will also be correlated with the pain and after a few trials, the light will be associated with the pain and will elicit an action. In pure ISO3, even if the light is removed for subsequent trials, its influence will always stay and when activated will always have the same effect.

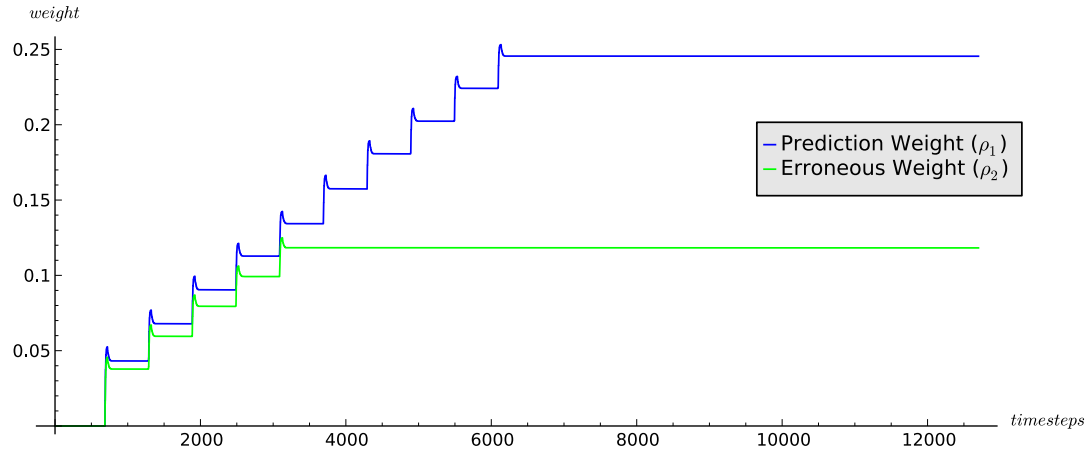


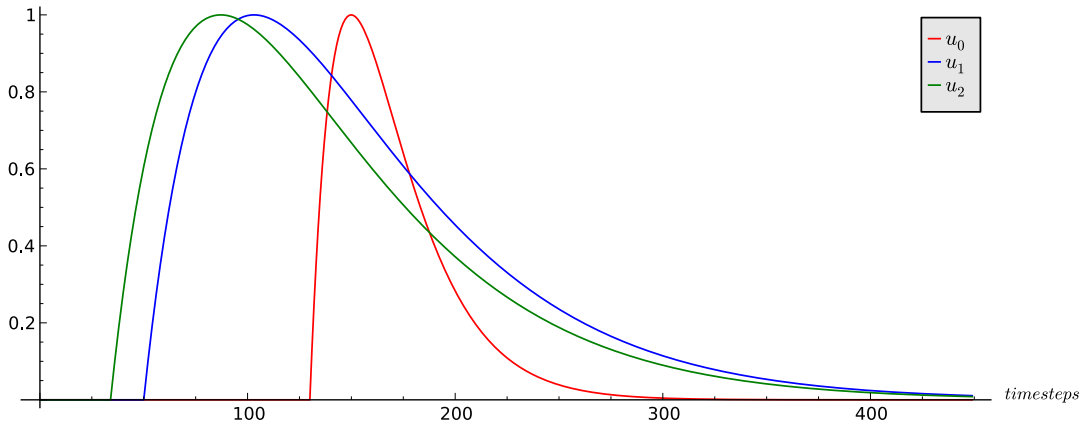
Figure 3.16: Spiking ISO3 weight trace. *Weight traces obtained when the spike sequence from figure 3.14 is used to train the system. The weights from all input paths stay stable after either their own or the reflex activity disappears.*

One mechanism which helps control the growth of weights and stimulate specialisation is used in a learning rule created by Oja [56].

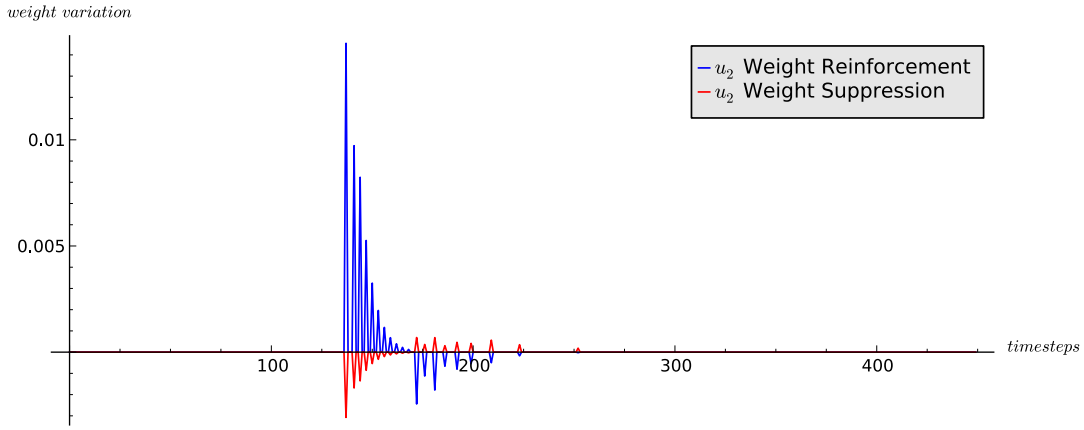
This rule was used as inspiration for an adaptation to the ISO3 learning algorithm. The Oja rule tries to control the weights of the synapses by using the postsynaptic activity to scale down the input weights. For the ISO3 adaptation, a synaptic competition [74] term was inserted as well. This term has the effect of allowing the strongest presynaptic input path to suppress the synaptic weight of the other input paths, thus allowing for a better differentiation. As can be seen in equation 3.12, the depression term is proportional to the postsynaptic activity variation and to the difference in activity between the most active pathway ($u_{max} = max(u_0, \ldots, u_n)$) and the pathway being updated (u_i). A new parameter ω was added as well in order to control the amount of influence of the new term. v' was chosen for the suppression term so that, as also happens with the reinforcement effect, the depression effect be dependent on how strongly or weakly the selected input correlates with a larger or smaller change of the output, therefore maintaining consistency in the learning rule.

$$\frac{d\rho_i}{dt} = \mu u_i v' \bar{R} - \omega(u_{max} - u_i)v' \tag{3.12}$$

An illustration of the weight evolution in different situations can be seen in figures 3.17 and 3.18.

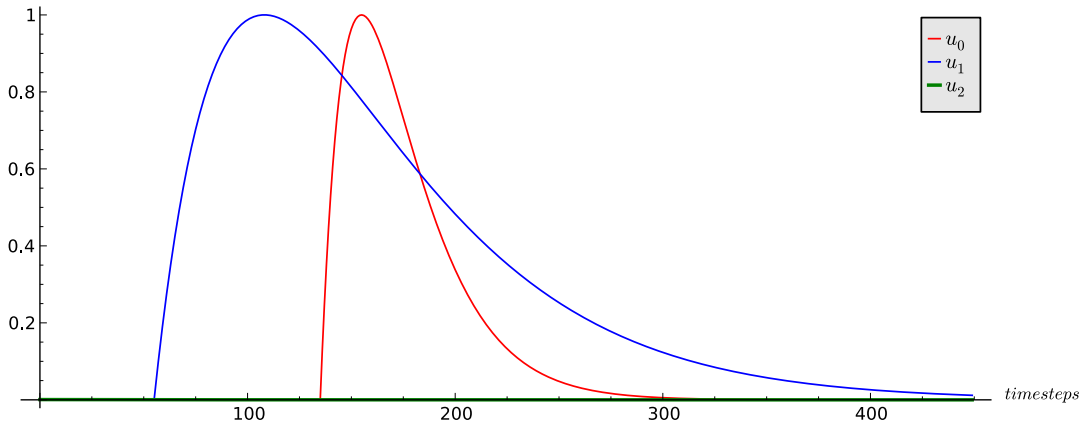


(a) Forgetful ISO3 Traces

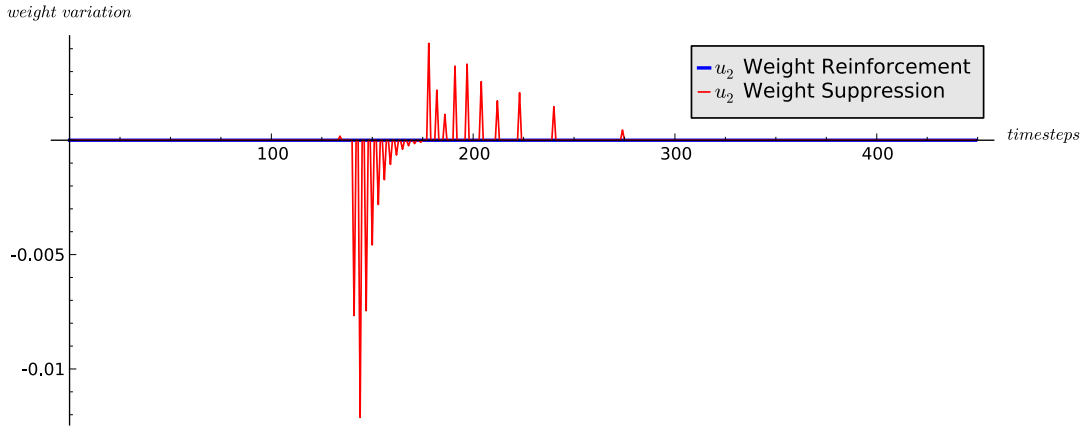


(b) u_2 weight variation components

Figure 3.17: Illustration of the change of weights in Forgetful ISO3. *In the presence of a stronger predictive signal (u_1), u_2 is slightly suppressed, but as its correlation with the output is strong, its net weight variation is positive.*



(a) Normal Walking



(b) Normal Walking

Figure 3.18: Illustration of the change of weights in Forgetful ISO3 in the absence of a signal. *When u_1 correlates with the output activity without u_2 being active, the weight of u_2 is strongly suppressed.*

Using this new weight change equation, a new set of open loop simulations was run. The input spike sequence was identical to that presented in figure 3.14.

For these tests, the following parameters were used:

- Reflex signal (x_0) band pass filter: $f = 0.008$ and $Q = 0.501$
- Predictive signal (x_1) band pass filter: $f = 0.003$ and $Q = 0.501$
- Erroneous signal (x_2) band pass filter: $f = 0.003$ and $Q = 0.501$
- $\mu = 0.2$
- $\omega = 0.2$
- The neuron was set to fire when the integration of the input traces reached 2.5
- All band pass filters were normalised to have a peak of value 1.0

Using the same example above to illustrate the new learning rule, after the light is removed in subsequent trials, with this new weight update rule, its weight will decrease with every trial as its importance relative to other, more consistent predictive inputs is decreased.

As can be seen in figure 3.19, with this new rule, the synaptic weight for the input path receiving the erroneous spikes is strongly suppressed by the predictive pathway while learning is still taking place, and continues to be lightly suppressed after learning has finished.

What this means is that, when learning finishes, the erroneous spikes will not cause as much disturbance as they would in the original learning rule. This disturbance will be further reduced with time as activity in the output of the postsynaptic neuron further suppresses its weight.

In order to further understand the behaviour of the new rule in response to random correlations, another simulation was performed, this time using, as the erroneous input, not a structured pattern, but random spikes (figure 3.21).

In the case of the adapted rule (equation 3.12), it can be seen that the weight is kept low due to the suppression present (figure 3.22).

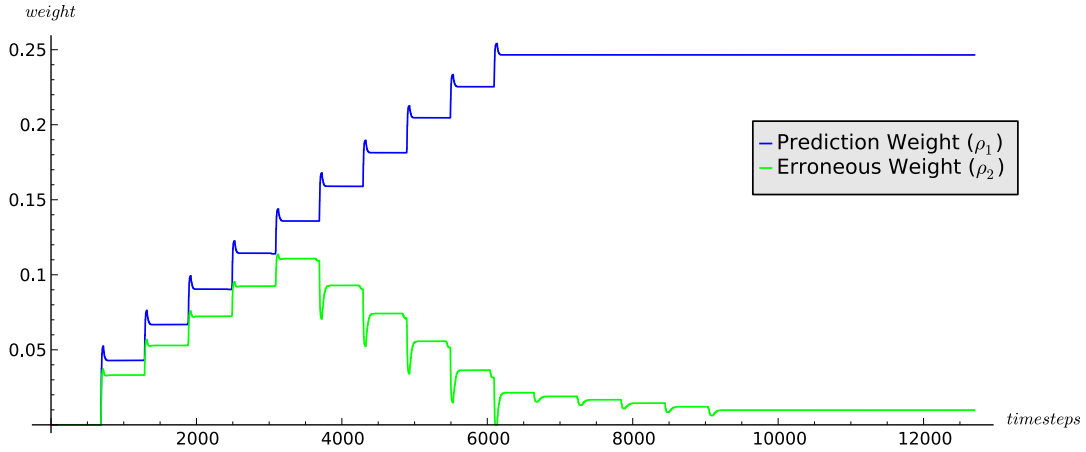


Figure 3.19: Forgetful ISO3 weight trace. *Weight traces obtained when the spike sequence from figure 3.14 is used to train the system. In the second part of the learning phase, when the extra input is removed, its weight decreases by a substantial amount due to the suppression performed by the predictive input at the same time as the output is very active due to the reflex action. This weight suffers another decrease after learning ends and the predictive path is active. This is a smaller decrease due to the smaller output activity.*

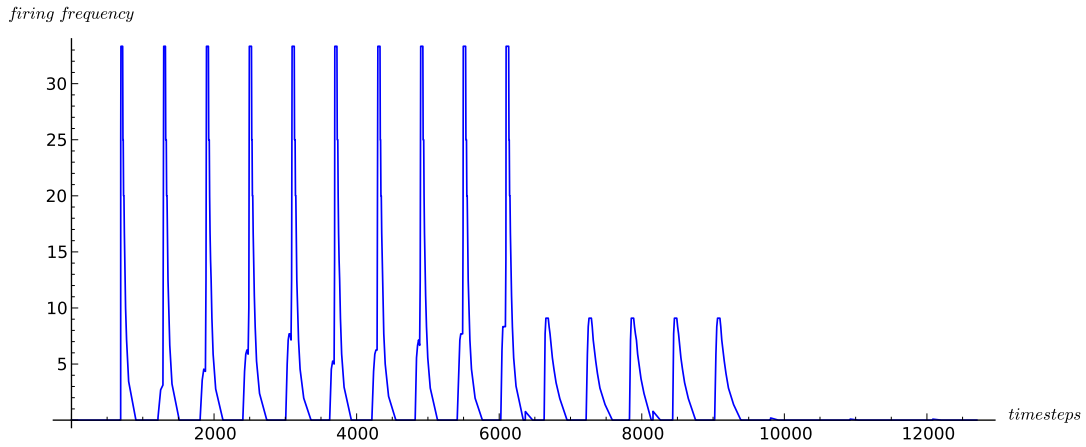


Figure 3.20: Output activity for the Forgetful ISO3 system. *Frequency plot of the ISO3 neuron output obtained when the spike train in figure 3.14 is used. It can be seen that there is no activity present when the erroneous spikes are presented after learning.*

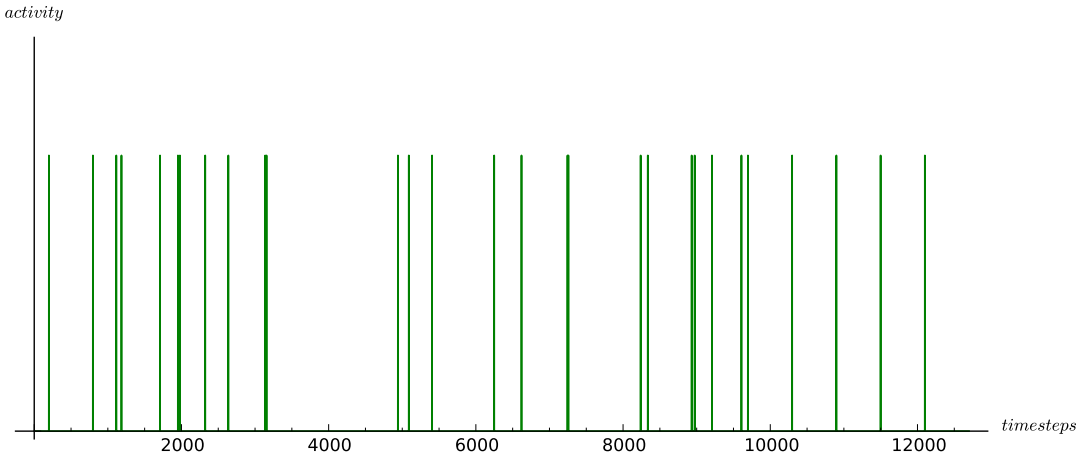


Figure 3.21: Random Input spikes *used for the noise robustness test.*

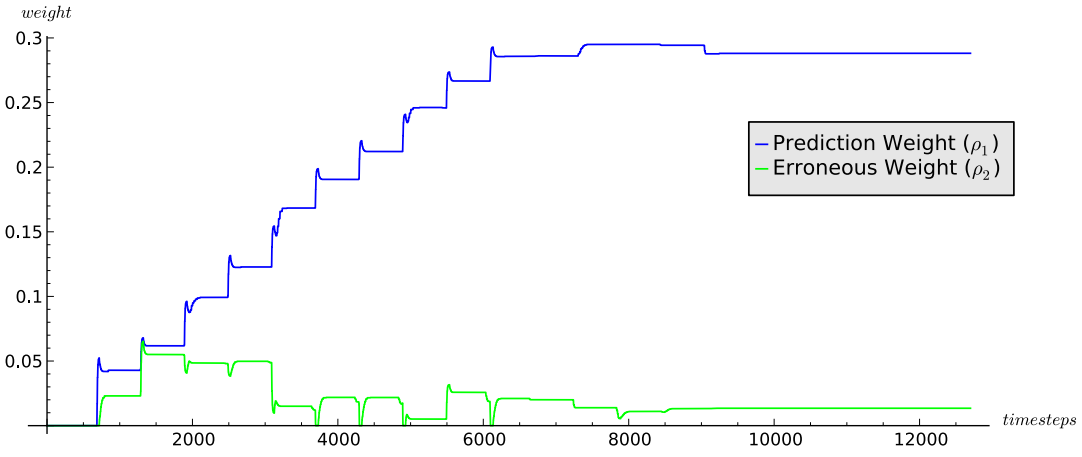


Figure 3.22: Weight traces obtained when the random spike sequence from figure 3.21 is used to train the Forgetful ISO3 system. *It can be seen that the weights of the erroneous input path are kept very low.*

When the erroneous spikes are used as the only input, the activity remains almost nonexistent due to the low weights (figure 3.23).

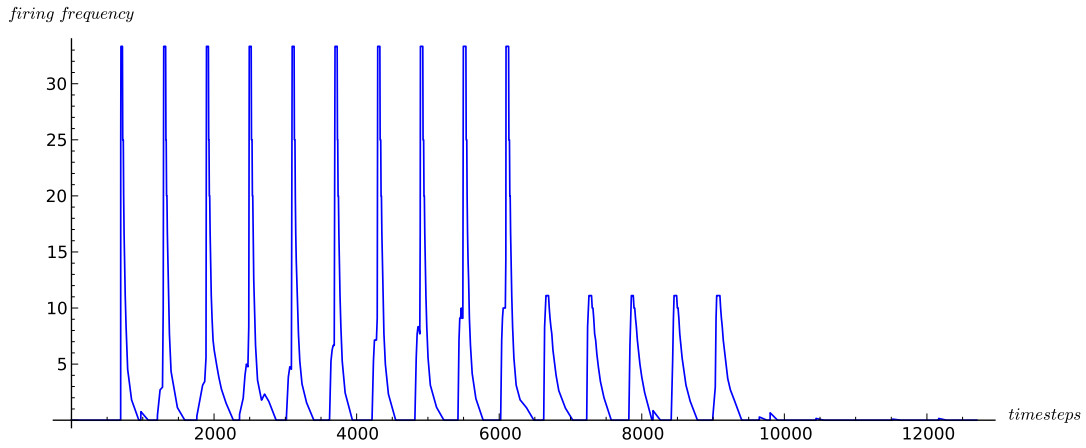


Figure 3.23: Frequency plot of the Forgetful ISO3 neuron output obtained when the random spike train is used. *It can be seen that there is no activity present when the erroneous spikes are presented after learning.*

The same simulation was performed with the Spiking ISO3 algorithm to allow a comparison with the results obtained with this new version. The weights grow as the erroneous inputs are correlated with the reflex actions and they continue that way after the reflex inputs are removed (figure 3.24).

Due to the higher weights achieved in the erroneous inputs, there is activity present in the end of the simulation, period when the erroneous inputs are the only ones with activity (figure 3.25).

3.5 Summary

In this Chapter, several types of reinforcement learning were presented, among them the ISO3 algorithm in which the learning system developed was based on. In a first version, the algorithm implemented was a spiking version of the ISO3 algorithm, thus named **Spiking ISO3**. This version seemed sufficient in the simple simulations performed, but when tested in the system simulations, it showed problems. These problems were related to its inability to "forget" which led to the incorrect increase of weights associated to inputs which were not important for the actions being learnt.

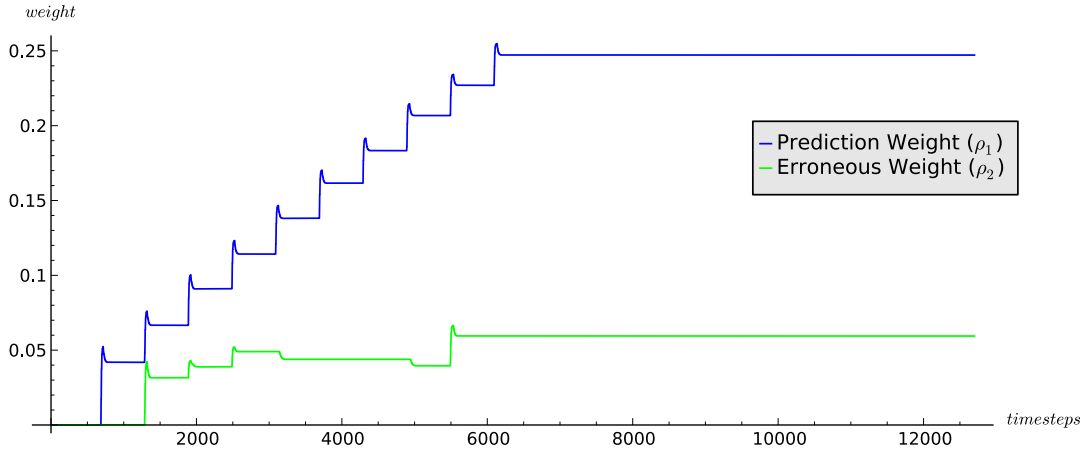


Figure 3.24: Weight traces obtained when the random spike sequence from is used to train the Spiking ISO3 system. *The weights for the erroneous input path grow with the occasional correlations and stay that way until the end of the simulation.*

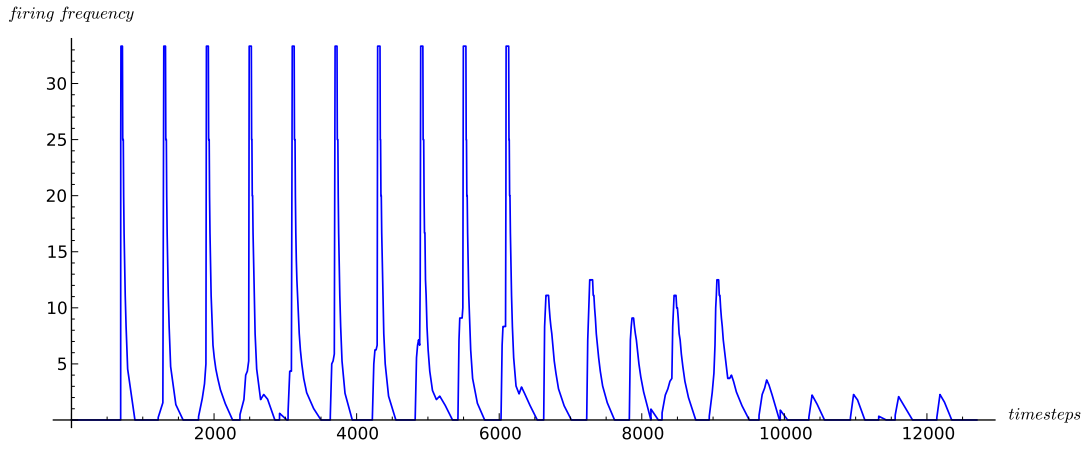


Figure 3.25: Frequency plot of the Spiking ISO3 neuron output obtained when the random spike train is used. *When the erroneous spikes are presented after all other activity has finished, it can be seen that there is some activity due to their accumulated weights.*

Any noisy activity during a reflex action would also increase the weights of the noisy pathways.

After the problem was identified, an alternative implementation was proposed, named **Forgetful ISO3**. In this second version of the algorithm an extra term was added to the weight variation calculation. This term was inspired in the Oja rule as the suppression was scaled by the output activity variation and besides that, it also allowed for the suppression of synaptic weights in weaker synapses by stronger ones, stimulating synaptic competition. It was then shown, by performing simulations using a model of the problematic situations, that this version behaved as expected.

Chapter 4
Vision processing and Depth Estimation

Vision is a sensing mechanism used by many animals as a mean to receive information about themselves and the surrounding environment. It is also used very often in robotics. It allows for longer distance sensing, coordination between entities, modulation of self movements, trajectory planning and many more useful activities [14,75].

A subsequent step is depth perception, which is usually derived from vision, although it can be performed through other means, for example sonar. Depth information allows for a better understanding of the physical structure of the surrounding world. It is used by animals in most of their activities, be it reaching for food, escaping from predators, moving in the environment or one of many others [76].

In robots it has the same importance, for object avoidance, grasping and other functions in which they have to interact with the surrounding environment. Similarly to other systems analysed in the thesis, depth estimation is one where nature has already provided many different ways of execution, depending on the evolutionary pressures each specie has suffered. There are many different cues used for depth perception. These can be divided into monocular and binocular clues. According to whether an animal is prey or hunter, nature has applied an evolutionary pressure in different directions [77]. “Prey” animals tend to have their eyes in opposite sides of their heads in order to maximise the field of view, which decreases the binocular overlap therefore increasing their dependence on monocular depth cues. “Hunters”on the other hand, need a better depth perception in order to be more successful in the attacks, thus rely more on binocular cues.

There are a variety of visual cues that contribute towards depth estimation, the main ones being: stereopsis, vergence, depth from motion and accommodation. The first two are applicable only to binocular systems, whilst the remaining ones are also applicable to monocular systems.

Stereopsis calculates depth by exploiting the parallax effect (difference in the projections of the objects) between the two slightly different points of view given by the eyes.

Vergence consists in using the angle of both eyes to triangulate the position of an object.

Depth from motion uses the characteristics of an object's projection movement in the retina in order to calculate its position in three dimensions.

Accommodation is the change in ciliary muscle activation used when focusing on objects at different distances, which require the lens in the eyes to change their focal length.

4.1 Depth from Motion

In the case of insects, on which this project took inspiration from for the development of the physical platform as well as the control system, there is evidence that motion parallax is used in depth estimation [78]. This form of depth estimation is also used by other animals, such as the macaque [79].

One of the project's intentions was to experiment with a "depth from motion" algorithm [80,81], which uses motion parallax to compute depth. This algorithm is based on the principle that, when moving at a constant speed towards a scene, the optical flow will only have a radial component and its value depends only on the specific position of the objects in the scene. As can be seen in figure 4.1, for points which have a projection in the same position of the retina, closer objects will have higher optical flow than further away objects.

By applying a network of edge sensitive neurons to the image on a radial manner (figure 4.2), it is possible to measure the time an edge takes to travel from one neuron to the next in the same axis, and, together with the projection position of the edge, to calculate the spatial position of those edges.

This allows a single camera system to generate a depth map, as long as its movement is always constant (i.e. constant direction and speed). In order to simplify calculations

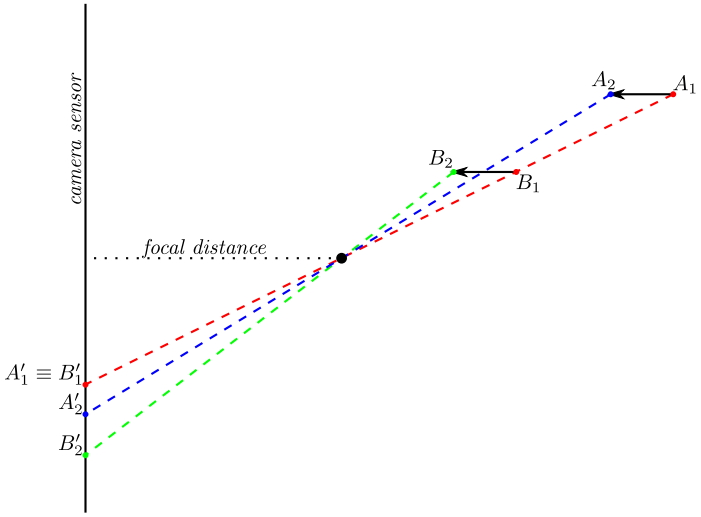


Figure 4.1: Difference in optical flow for two different points. *Points which have the same projection but different distances to the camera will create different amounts of optical flow when their distance to the camera is reduced. In this figure, points A and B start at positions A_1 and B_1 respectively. These two position have projections A'_1 and B'_1, which are coincident. A certain horizontal movement from the camera can be represented by the decrease in distance from the points with regard to it. After this displacement, it can be seen that the point which is closest, B, generates a projection in the camera sensor further away from the original projection than the one from point A, which is at a greater distance from the camera, thus B creates a stronger optical flow. This characteristic is what allows this type of system to generate a depth map from a single camera moving through a scene.*

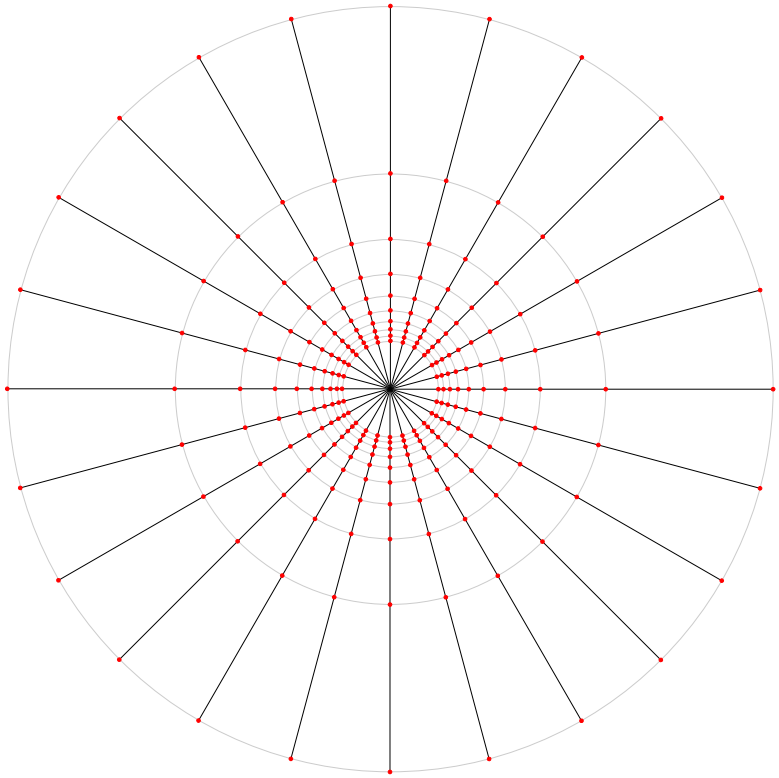


Figure 4.2: Neuron Distribution for the depth from motion algorithm.

and the analysis of this movement, the neurons were laid out in a way that makes the time an edge takes between activating two consecutive neurons constant throughout the axis (figure 4.3).

A prototype analog VLSI (Very Large Scale Integration) version of this algorithm had already been developed by the research group [82–87] which was successfully tested with artificially generated inputs and with simple image sequences. Due to those promising results, a more complex and higher resolution chip, implementing the same algorithm, was to be designed by a subset of the original authors.

This hardware system was to be connected with a camera which mimics the behaviour of a retina [44]. This camera has an AER[1] output, thus theoretically allowing its spike stream to be directly fed to the "depth from motion" chip.

[1] AER or Address Event Representation is a communication protocol, popular among the neuromorphic community. It consists in a mechanism which converts a spike into the address of the neuron responsible for the spike. This address is then checked on a routing table and it is then sent to the corresponding neurons in the other systems, where this address is then converted into a spike again. This is an asynchronous system, which mimics the spike transmission through synapses.

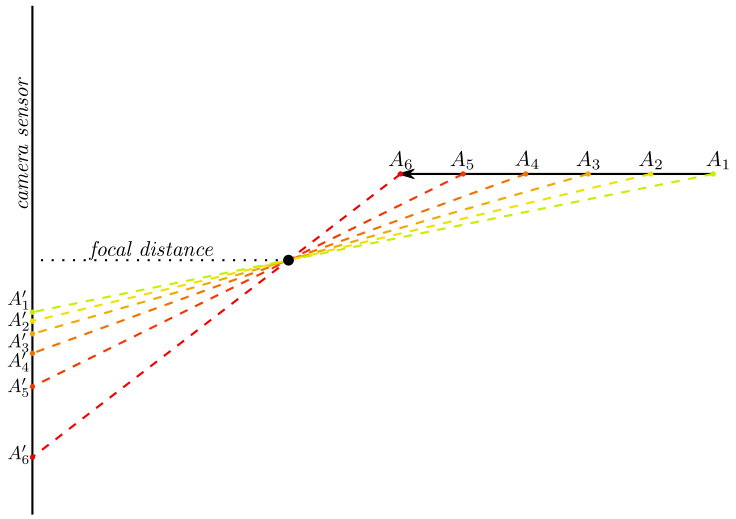
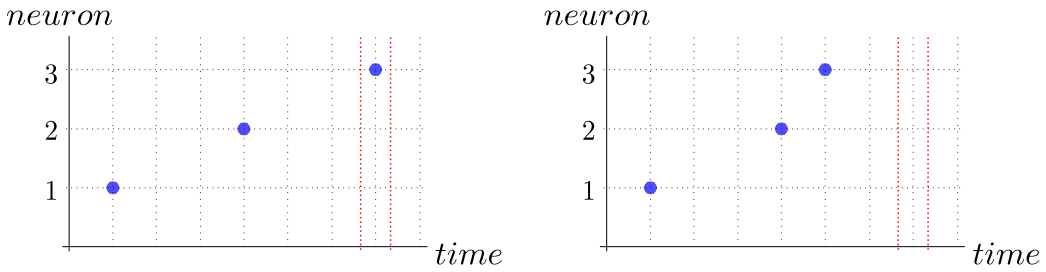


Figure 4.3: An edge projection propagating through the camera sensor. *For a constant shortening in the horizontal distance between an edge A and the camera (caused by the camera moving with a constant speed towards the edge), the distance from one position of its projection A' to the next increases.*

One other important characteristic of the algorithm implemented by the chip was the presence of a noise filtering algorithm. This was also described in the original papers and it consisted in a validation step performed for each edge before its value was sent to the output. This was done by taking advantage of the way the neurons were spaced in the retina, which guarantees (when the speed is constant) that the projection of edges will take the same amount of time to travel between any two consecutive neurons from the same radial axis. The time an edge takes between two neurons is used to predict the time of arrival of the same edge to the next neuron in the series. In case this prediction is proved correct within a certain degree of accuracy, the edge value is validated and sent to the output. If an edge does not arrive to the next neuron in the expected timeframe, for example if one of the spikes was caused by noise or the edge never gets to the next neuron, it will be discarded as erroneous (figure 4.4). This noise filtering system guarantees a certain quality of the values returned, as no edge will have a depth value until it has been confirmed by two consecutive measurements.

After the design of the hardware was finalised by the team, the details of its implementation were used to start the development of an accurate software model to be used in preparation for the integration with the retina and the learning systems.



(a) Valid sequence **(b)** Invalid sequence

Figure 4.4: Example of filtering algorithm activity. *When the edges are detected by consecutive neurons, the times it took between them is used to predict the time it will arrive to the next neuron in the axis. If it arrives within the expected time window (figure **a**) the edge is validated and its depth value is sent to the output of the system, otherwise, if the timing does not match within a certain tolerance (illustrated by the red vertical bars), this edge will not be validated and therefore will not be sent to the output (figure **b**).*

4.1.1 Experiments

Figure 4.5: Koala robot *[88]. A Koala robot was used as a platform to mount the artificial retina. As it is wheeled, the experiments are easier to setup and provides a more stable platform for the camera.*

As a first test, the retina camera was mounted on a Koala robot (figure 4.5) and a series of runs was performed and recorded. On these runs, several thin boards were placed in the ground, perpendicular to the robot motion (figure 4.6). These recordings were then used to verify the response of the "depth from motion" chip model. For the test of the model, only one axis of neurons was used. The axis used was the one placed at the 6 O'Clock position (figure 4.7).

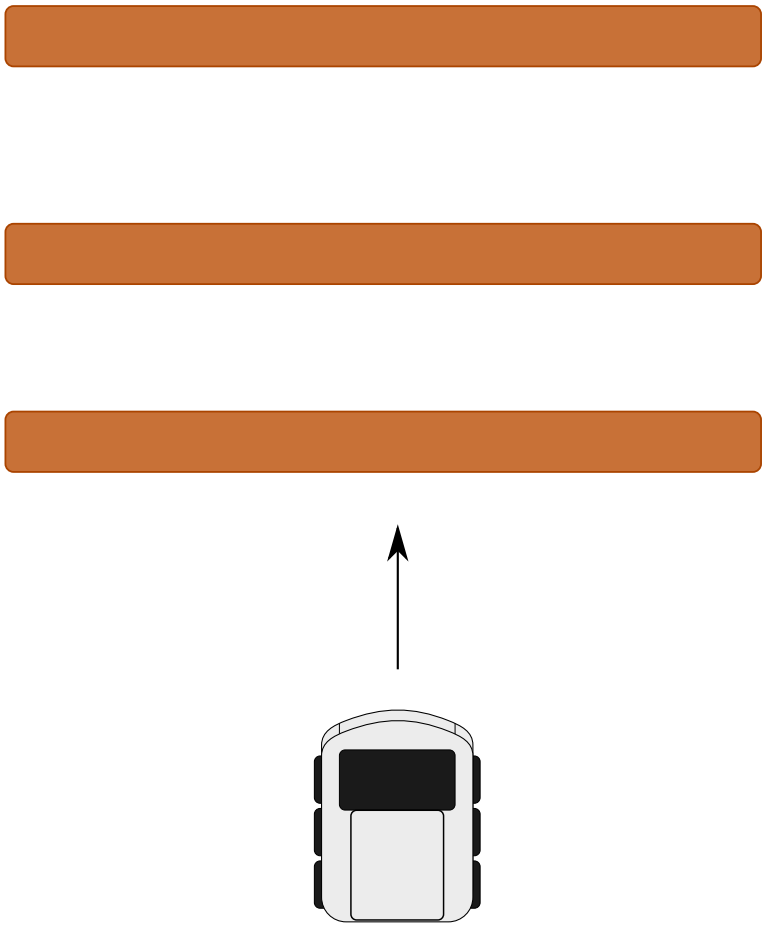


Figure 4.6: Koala experiment. *The Koala robot, with the camera mounted on top, was put in forward motion towards a set of thin boards placed on the ground perpendicularly to the direction of movement.*

Figure 4.7: Neuron axis selected for the analysis of the depth from motion algorithm.

Input Noise

One of the problems seen was a lack of nearly any output from the system when fed the data from the artificial retina. This turned out to be caused by the combination between the noise filtering approach and the characteristics of the sensor used. The artificial retina has some background noise which causes the appearance of false edge detections. Another characteristic of the sensor is the creation of a sequence of spikes with variable length, rather than a single event, for each edge that is actually detected. It was found that under these conditions, the noise filtering system has a severe problem, if there are many false edges being fed to the system, they will cancel correct calculations by activating one of the three neurons in the sequence at a wrong time, which means that when the system compares the times between the steps, these will be too different and therefore the whole series will be considered invalid. This effectively causes a denial of service on the depth from motion chip.

Several simple filtering techniques were applied to the data, the requirement for these was that no significant hardware was to be used between the camera and the sensor, as the goal of this architecture was to simplify a usually complex process. If the solution to this problem was more complex than the system itself, it would defeat its purpose. The first attempt at improving the system consisted of eliminating events that are not confirmed within a certain time window by others within the surrounding

region. This relies on the fact that edges are usually larger than a single pixel, so it should be detected by the neighbours as well. This proved helpful at eliminating some background noise by reducing the total amount of spikes by up to approximately 60% before valid edges started being filtered out as well, but some events were still present and it was not effective against the spike trains generated by real edges which would extend until after the edge had reached the next measuring point in the axis, thus invalidating any time interval measurement between them.

On a second attempt, the resulting spikes obtained above, were then filtered again, by applying a refractory period, which partially eliminates the repeat firing caused by an edge. The problem with this approach was that the duration of the burst was variable, some spike trains lasted up to 5 times longer than others, even making it difficult in some cases to understand when one edge stopped and another started. This made it difficult to set the value for the refractory period, so if the refractory period was too long, it would affect subsequent edges, and if it was too short, the left over spikes would cancel the whole calculation again.

After these filters were applied, the system started outputting more values, usually only 1 to 3 depth values per edge were returned during the whole run, which were not enough for the required purpose. These filters were tried on pre-recorded data, which allowed for the timings of the events to be maintained, but implementing them on real time data would affect the timings in a way which would be detrimental to the depth calculations.

Movement effects

Another problem with this setup was detected after viewing image sequences captured in the simulator described in section 5.2. The simulator allowed the prediction of what would happen in terms of camera movement in the real setup. Using a simple walking controller, it could be seen that a severe amount of movement was induced in the camera, which could have an impact on the depth from motion algorithm, as this relies on a steady forward motion without any other components. One of the types of movement detected was a roll, which happened when the support of the weight was transferred from side to side. An illustration of this can be seen in figure 4.9. This type of movement affects all neuron axis in the same way, as the only component is

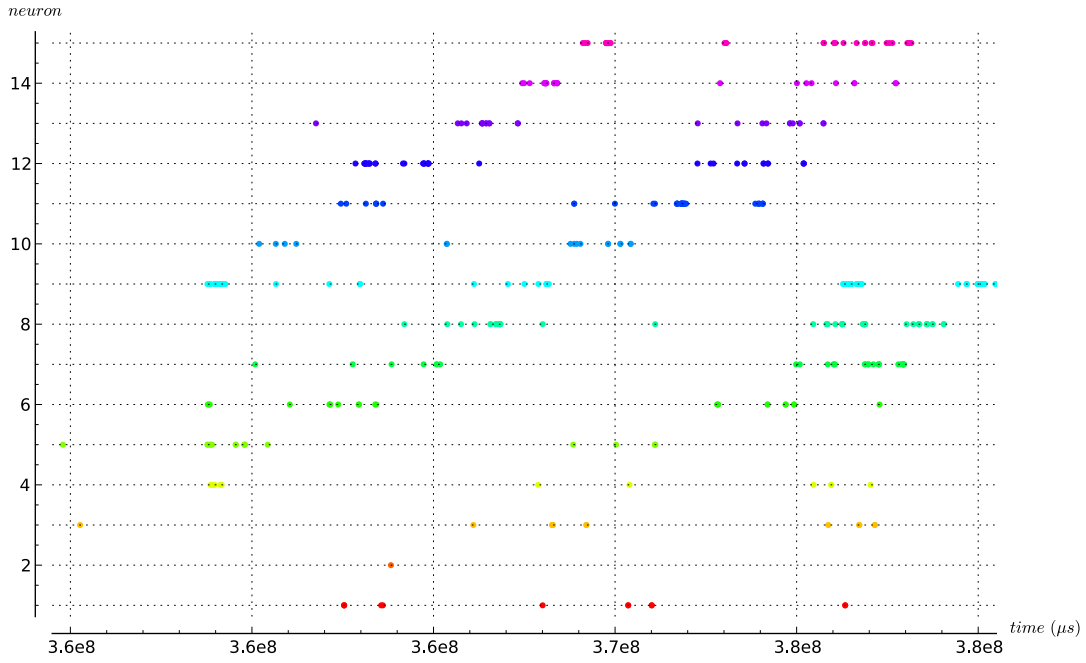


Figure 4.8: Edge Events. *Raw events received from the neurons situated at the 6 O'Clock axis when moving towards a series of white lines laid out on the ground. This test was performed with 15 neurons, the spikes of which are represented by the dots in the graph, with each colour corresponding to a different neuron. Neuron 1 was close to the centre of the image, whilst neuron 15 was towards the periphery.*

perpendicular to the directions of the axis. It might cause an edge to passe around the neurons, therefore causing gaps in the data, which extend to the next neuron as well, due to the way the error detection mechanism works.

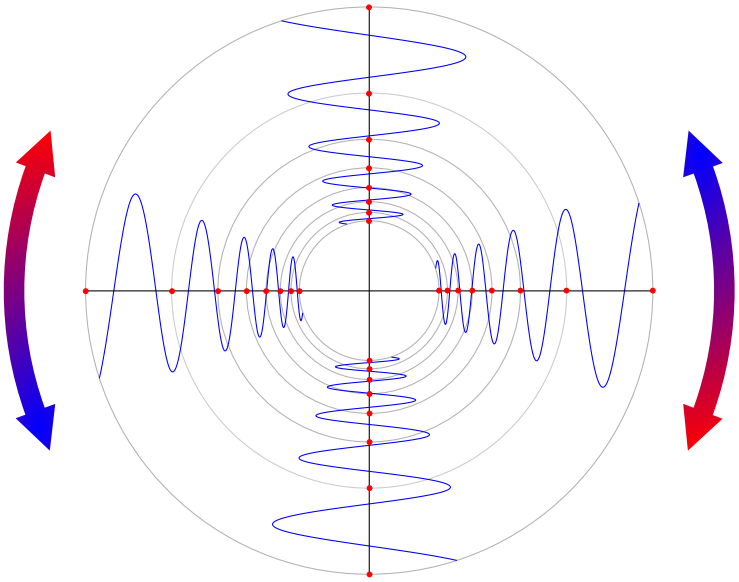


Figure 4.9: Tracking of different edges along 4 different neuron axis in the presence of a cyclic roll movement in the camera. *This type of movement affects all of the neuron axis equally. Depending on the amplitude of this movement, it might cause problems tracking the edges, as their projections will at times skip neurons in the axis by going around them.*

The other type of movement was a change in pitch, which caused the camera to go up and down, as the weight shifted back and forward. This component is illustrated in figure 4.10. In this case, there can be two components of movement with regard to the axis, it can be perpendicular to the axis, therefore causing a similar problem to the one described above, and it can be parallel to the axis, causing a change in the velocity of the edge projection in the camera sensor, thus affecting the calculations for these neurons.

In order to counteract the effect of small movements occurring perpendicularly to the axis such as in the case of the roll movement, instead of single pixels being used to activate the corresponding neurons in the axis, a section of the circle containing the neurons could be used (figure 4.11). As long as the forward speed was still constant, this would have little impact on the depth calculation.

This change would not fix the movement occurring in the direction of the axis though,

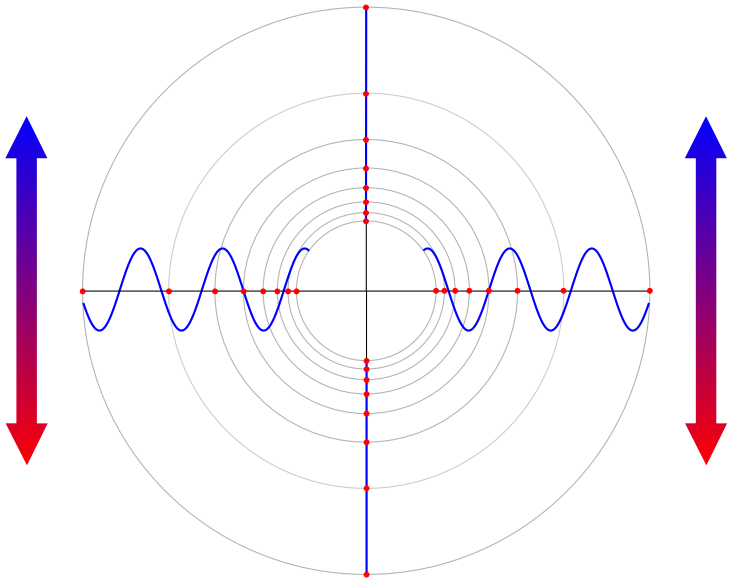


Figure 4.10: Tracking of different edges along 4 different neuron axis in the presence of a cyclic pitch movement in the camera. *This type of movement affects the axis in different ways, when the movement is parallel to the direction of the axis, it does not affect the projection shape, but its velocity along the sensor. When the movement is perpendicular to the axis, it will cause a similar problem to the roll, i.e. the projection will not activate some neurons in the axis. In the remaining axes, both components are present, with varying degrees.*

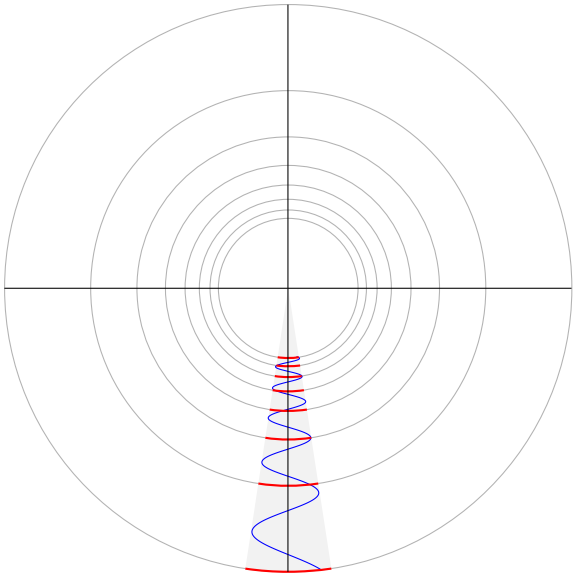


Figure 4.11: Possible way to counteract the effect caused by movement perpendicular to the neuron axis. *By using a series of pixels, all at the same distance from the centre of the sensor instead of a single pixel, the edge detection can still work for low amounts of rotational movement.*

so in order to clarify this effect, and because it is also useful to measure how sensitive the system is to accurate readings of the velocity at which edges propagate in the field of view, a mathematical analysis was performed on the system with the purpose of quantifying how much impact this has on the algorithm. For simplicity, a pin-hole camera model is used and only a 2 dimensional approach is taken as any other situation can be easily translated into it.

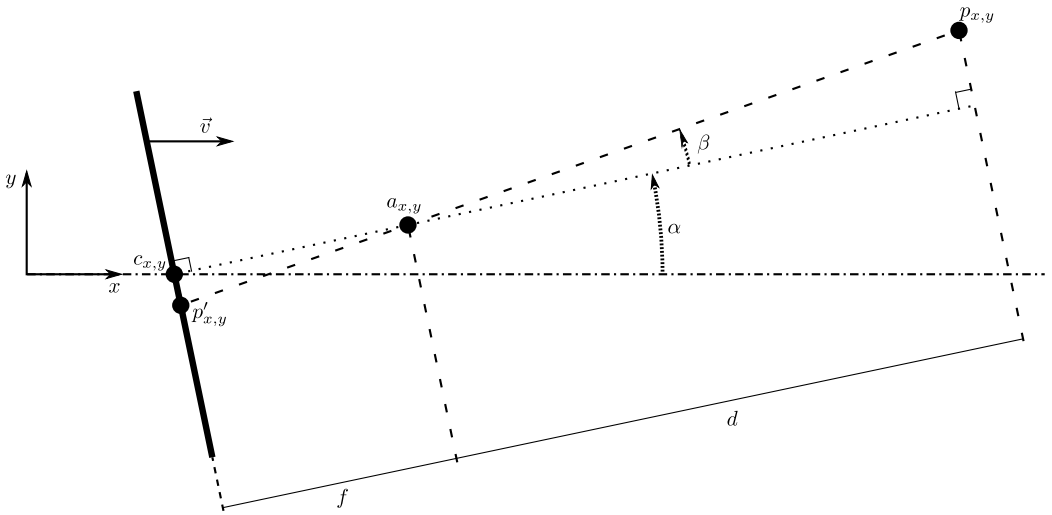


Figure 4.12: All the angles and distances used in the depth calculation. *c is the centre of the image sensor, p is the point being tracked, p′ is the projection of that point in the image sensor, and a is the focal centre of the camera. α is the angle of the camera, β is the angle of the sightline that passes through point p in relation to the centre sightline. f represents the focal distance and d represents the distance between the camera and the plane containing point p.*

Figure 4.12 represents all the angles involved in the calculation of the position of a point in space on the camera sensor. There are 3 main points in the diagram, point $c_{x,y}$ represents the centre of the sensor, which is assumed to start at the origin $(0, 0)$, point $a_{x,y}$ represents the position of the camera aperture and point $p_{x,y}$ is the point in space being analysed. The 2 main angles to pay attention to are α, which represents the angle of the sensor, and β, which represents the angle between the perpendicular of the sensor with the line formed by points p and a. f is the focal length of the camera (distance between the centre of the sensor to the aperture). r is the distance between the centre of the sensor and the projection point p'.

The equations that describe the static state of the system are:

$$r = f \tan(\beta), \tag{4.1}$$

with

$$\beta = \arctan\left(\frac{p_y - a_y}{p_x - a_x}\right) - \alpha, \tag{4.2}$$

$$a_x = f \cos(\alpha) + c_x, \tag{4.3}$$

$$a_y = f \sin(\alpha) + c_y, \tag{4.4}$$

When considering a system moving in the x axis with the camera plane perpendicular to the direction of movement as described in the depth from motion algorithm, we add the time variable t to the equations. The scalar velocity of the projection p' is represented by v'_p. In order to simplify the equations, the camera (point c) is considered to start at position $(0, 0)$ so the following is obtained:

$$a_x(t) = f \cos(\alpha(t)) + c_x(t), \tag{4.5}$$

$$a_y(t) = f \sin(\alpha(t)) + c_y(t), \tag{4.6}$$

$$\beta(t) = \arctan\left(\frac{p_y - a_y(t)}{p_x - a_x(t)}\right) - \alpha(t), \tag{4.7}$$

$$r(t) = f \tan(\beta(t)), \tag{4.8}$$

$$v_{p'}(t) = \frac{dr}{dt} \tag{4.9}$$

The depth from motion algorithm assumes the following constraints:

$$c_x(t) = vt, \tag{4.10}$$

$$c_y(t) = 0, \tag{4.11}$$

$$\alpha(t) = 0, \tag{4.12}$$

where v represents the scalar velocity of the camera.

By substituting equations 4.10, 4.11 and 4.12 in equations 4.5 and 4.6, the following is

obtained:

$$a_x = f + vt, \tag{4.13}$$

$$a_y = 0, \tag{4.14}$$

Which gives:

$$\beta(t) = \arctan\left(\frac{p_y}{p_x - f - vt}\right), \tag{4.15}$$

and subsequently:

$$r(t) = \frac{f p_y}{p_x - f - vt} \tag{4.16}$$

$$v_{p'}(t) = \frac{f v p_y}{(p_x - f - vt)^2} \tag{4.17}$$

and because

$$p_x - f - vt = d \tag{4.18}$$

the following general equation is obtained:

$$v_{p'}(t) = \frac{f v p_y}{d^2} \tag{4.19}$$

In the ideal scenario, $\alpha = 0$, which would mean

$$\begin{aligned} \frac{p_y}{d} &= \frac{r}{f} \\ p_y &= \frac{dr}{f} \end{aligned} \tag{4.20}$$

which gives the final equation used for the calculation of the depth given the horizontal velocity is constant:

$$\begin{aligned} v_{p'} &= \frac{vr}{d} \\ d &= \frac{vr}{v_{p'}} \end{aligned} \tag{4.21}$$

With these equations, it is now possible to simulate what happens when the camera is no longer fixed. For these tests, small oscillatory movements are applied to the camera, simulating the effects of the camera movement when assembled on the robot.

The deviation of the measurements compared to the stable camera scenario can then be measured. For the tests, a simple sinusoidal wave was used to model the variation of the angle α.

$$\alpha\left(t\right) = a\cos bt, \tag{4.22}$$

Where a is the amplitude of the movement and b controls the frequency.

A similar function (equation 4.23) was also used to model the vertical displacement of the camera.

$$c_y\left(t\right) = g\cos ht, \tag{4.23}$$

Using the equations above, the distance of the point projection to the centre of the sensor (r) takes the form of equation 4.24.

$$r\left(t\right) = f\tan\left(-a\cos\left(bt\right) + \arctan\left(\frac{f\sin\left(a\cos\left(bt\right)\right) + g\cos\left(ht\right) - p_y}{f\cos\left(a\cos\left(bt\right)\right) + tv - p_x}\right)\right), \tag{4.24}$$

$v_{p'}\left(t\right)$ can now be calculated as well through the derivation of equation 4.24 which results in equation 4.25.

$$\begin{aligned} v_{p'}\left(t\right) = &\left(\tan\left(-a\cos\left(bt\right) + \arctan\left(\frac{f\sin\left(a\cos\left(bt\right)\right) + g\cos\left(ht\right) - p_y}{f\cos\left(a\cos\left(bt\right)\right) + tv - p_x}\right)\right)^2 + 1\right)\cdot \\ &\cdot\left(ab\sin\left(bt\right) - \frac{\frac{abf\sin(bt)\cos(a\cos(bt)) + gh\sin(ht)}{f\cos(a\cos(bt)) + tv - p_x}}{\frac{\left(f\sin(a\cos(bt)) + g\cos(ht) - p_y\right)^2}{\left(f\cos(a\cos(bt)) + tv - p_x\right)^2} + 1} + \right. \\ &\left. + \frac{\left(abf\sin(a\cos(bt))\sin(bt) + v\right)\left(f\sin(a\cos(bt)) + g\cos(ht) - p_y\right)}{\left(f\cos(a\cos(bt)) + tv - p_x\right)^2}}{\frac{\left(f\sin(a\cos(bt)) + g\cos(ht) - p_y\right)^2}{\left(f\cos(a\cos(bt)) + tv - p_x\right)^2} + 1}\right) f \end{aligned} \tag{4.25}$$

With equations 4.24 and 4.25, the value of the distance as calculated by the depth from motion system being analysed can be obtained by replacing them in equation 4.21.

For the following test, the point in the scene was initially set $1m$ away from the camera ($p_x = 1$), $10cm$ below the camera axis ($p_y = -0.1$) and the speed of the robot was set

to $5cm/s$ ($v = 0.05$). The extra movements of the robot were defined as having both rotational and vertical displacement components. The frequency of both the camera angle oscillation and the vertical displacement was set to $0.5Hz$ ($b = h = \pi$), the angle variation amplitude was set to 1 degree ($a = 0.5$) and the amplitude of the vertical motion was set to $1cm$ ($g = 0.005$).

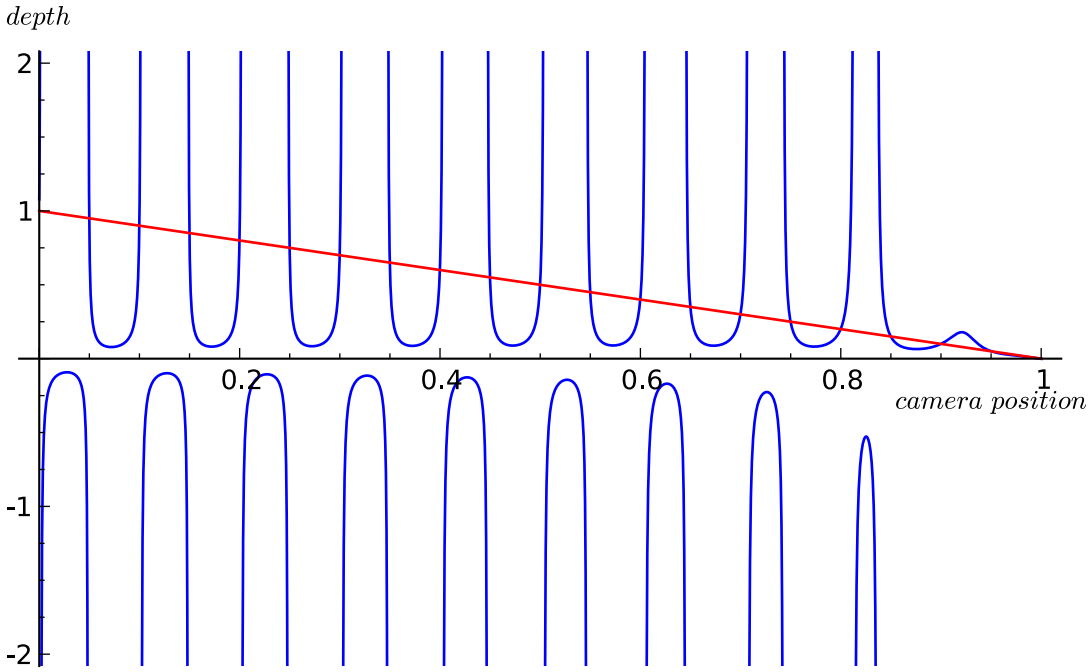


***Figure 4.13: Comparison between the calculated values for the depth in the two cases**, stable movement (red) and unstable movement with both rotational and translational components (blue). As can be seen, the measured depth values in the unstable case have almost no trace of what the real depth should be, apart from when the camera is already really close to the point. The reversal in the depths at some points are due to the fact that the movement causes the projection of the point to starts moving toward the centre of the camera, which is the opposite of the expected direction.*

As can be seen from figure 4.13, a slight wobble in the camera movement causes very big problems in the depth calculation. This error decreases the closer the point is to the camera, but the usable range is quite small.

Figure 4.14 shows how the error evolves for the same test, but with changing variations in angles. It can be seen that the error quickly becomes a major problem as the movement of the camera becomes more significant.

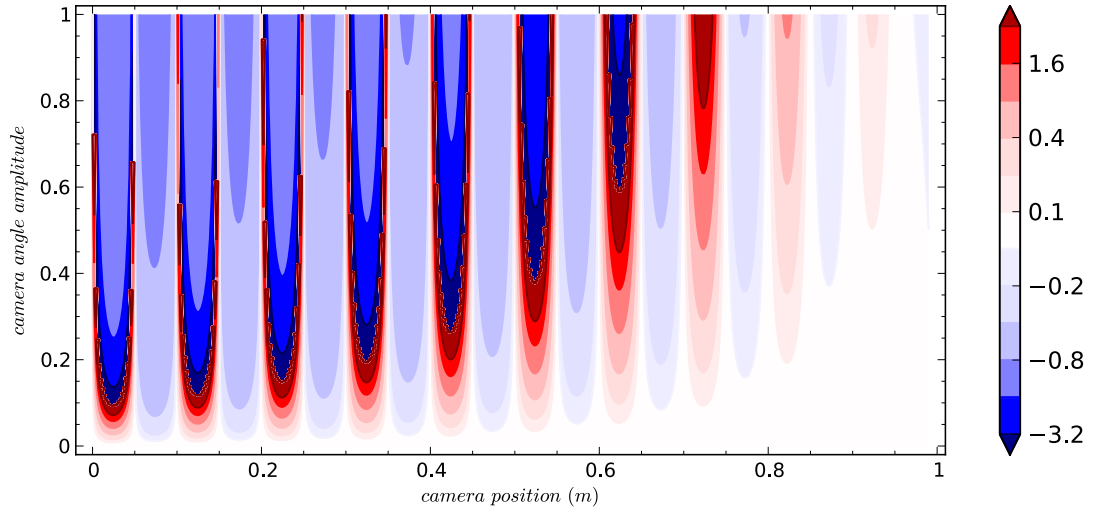


Figure 4.14: Contour plot of the error ratio between the unstable movement, with both rotational and translational components, and the stable one. *It can be seen that increasing the amplitude of the movement (y axis), the causes the error in the measurements to increase very fast to the point of unusable.*

A rotational movement can be relatively easy to counteract using a pan and tilt mechanism controlled by a gyroscope, but the translational movement caused by the up and down movement of the robot is harder to correct without having an overly complex hardware stability system or without having to change the depth from motion system itself. Due to this fact, another simulation was performed considering only the vertical translational movements caused by the robot ($\alpha = 0$). In this simulation, the translation movement was maintained, but the rotation movement was eliminated, i.e. the camera is always parallel to the ground, but the vertical movement maintains the same characteristics as the previous simulation. In this case, the distance to the centre of the sensor (r) adopts the following equation

$$r(t) = \frac{(g\cos(ht) - p_y)f}{tv + f - p_x}, \tag{4.26}$$

which, when derived, gives equation 4.27 for $v_{p'}$.

$$v_{p'}(t) = \frac{dr}{dt} = -\left(\frac{gh\sin(ht)}{tv + f - p_x} + \frac{(g\cos(ht) - p_y)v}{(tv + f - p_x)^2}\right)f, \tag{4.27}$$

Plotting equation 4.21 using $r(t)$ and $v_{p'}(t)$ from equations 4.26 and 4.27, figure 4.15 is obtained.

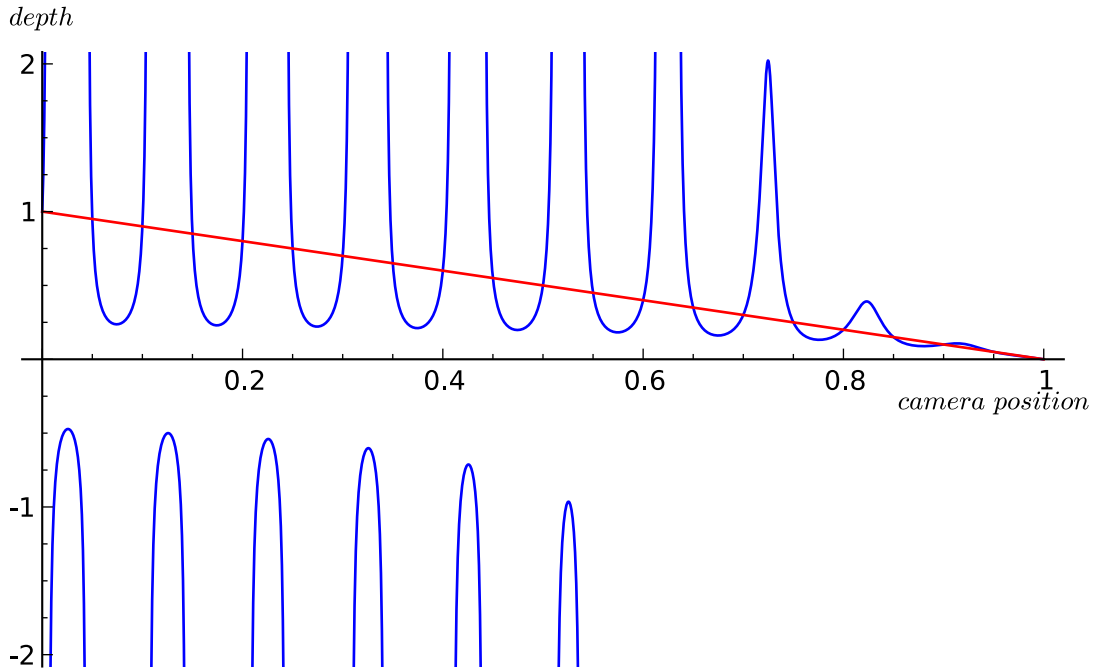


***Figure 4.15: Comparison between the calculated values for the depth in the two cases**, stable movement (red) and unstable movement due with a translational movement only (blue). The translational movement has 0.5Hz and has an amplitude of 1cm. As can be seen, the depth values are still extremely degraded by the movement and, although slightly better than having rotational movement as well, it is still unusable until the object is already very close.*

As can be seen from figure 4.15, the elimination of the rotation movement barely changes the error. In fact, by examining figure 4.16, which represents the contour plot of the error induced by a translation movement ranging from 0 to 1cm in amplitude, it can be seen that in order to obtain less than 10% of error at 30cm from the point, in the circumstances tested, the amplitude of the movement cannot be greater than 1mm.

4.1.2 Summary

Even though the depth from motion algorithm implementation analysed in this chapter works in theory [84], and it has been verified to work in practise under very con-

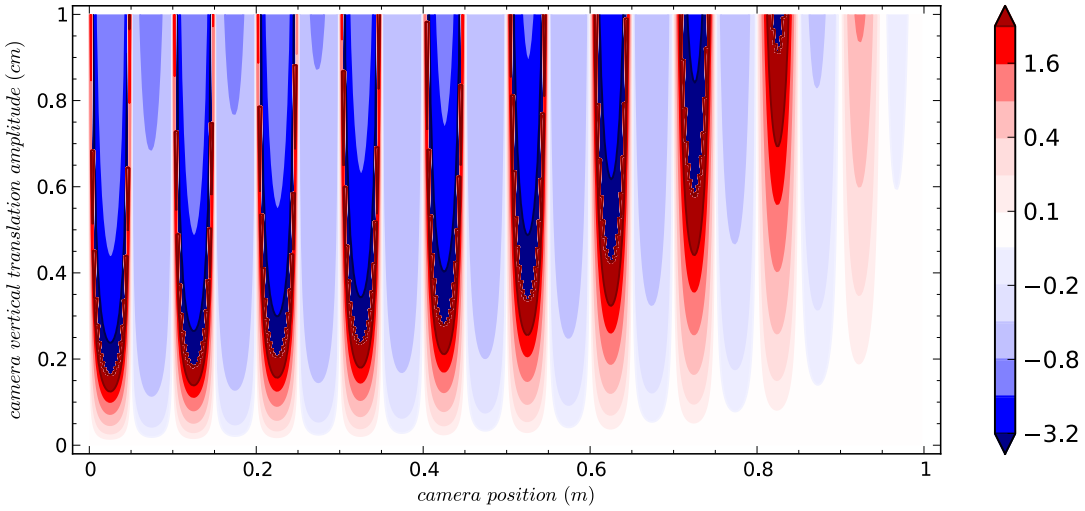


Figure 4.16: Contour plot of the error ratio between the translation movement and the stable one. *It can be seen that, as with the previous case, the error still increases quickly to unusable levels.*

trolled circumstances [85, 86], the system, as it was implemented, revealed several shortcomings.

The first problem observed was caused by the excessive amount of noise coming from the artificial retina which, when fed to the model of the depth from motion chip, made it difficult to obtain enough depth data. The design of the chip took into account a certain noise robustness, by discarding unconfirmed depth values, but this naive approach meant that when there is too much noise, the erroneous edge spikes will essentially cancel the previous correct calculations and will cause it not to output as many depth values as needed to perform a valid scene analysis.

The second problem observed was that the camera movement caused by the robot locomotion was shown to impact heavily on the calculations necessary to perform depth from motion and even when cancelling all the rotational components of the movement, the remaining translational component was shown to also cause large errors. This led to the conclusion that this algorithm is not suitable for applications in which the optical flow field caused by the camera movement is not perfectly radial.

Even if the noise problem could be decreased, by the use of filtering mechanisms, the algorithm would still not perform adequately due to the movement of the robot.

This led to the decision to abandon this system and adopt another, more robust and proven, biologically inspired depth calculation system.

4.2 Stereo Vision

Stereo vision works by using two slightly different points of view and detecting the difference in position of the points in the scene. This difference is called binocular disparity and the higher it is, the closer the object is located. A very interesting review of this concept together with more in depth information can be found in [89]

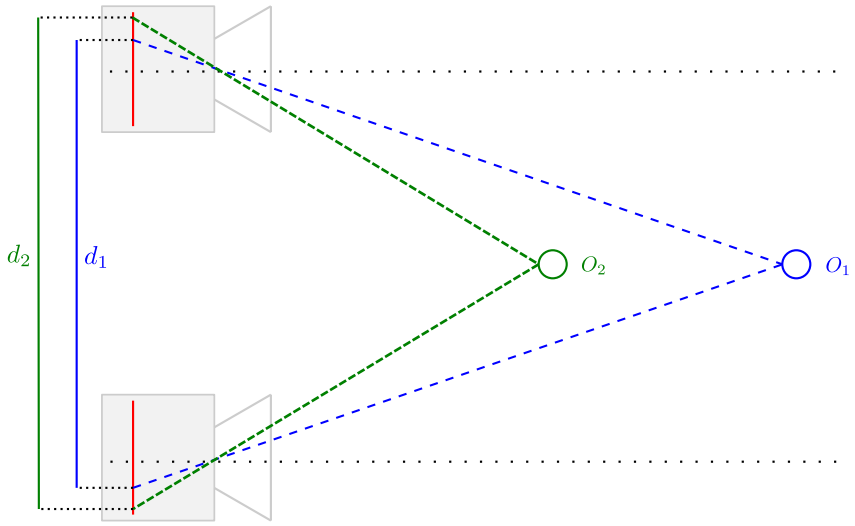


Figure 4.17: An illustration of Binocular Disparity. *The closer an object is, the larger the different between the positions of its projection in the retina. In this example, object 1 (O_1) is further away than object 2(O_2) so its disparity (d_1) is also smaller than that of object 2 (d_2).*

As a point's distance approaches the convergence point between the cameras, its disparity will approach zero. In this case the cameras being used have the same orientation, so the convergence point lies at infinity, which simplifies the calculation, as in this situation the disparity will only have one sign and it is a monotonic function of the distance between the point and the camera. The closer a point is to the cameras, the larger the difference between its projections on both cameras. This mechanism can be used to reconstruct a 3D model of the scene by analysing the difference in projection coordinates of the features composing the scene.

Most stereo systems rely on frame based cameras but, during the writing of this

thesis, some advances have been made on stereo calculation based on the artificial retinas described in the previous section [90].

4.2.1 Implementation

This system was originally implemented using a set of 2 SRV-1 boards, each containing a Blackfin® processor (figure 4.18).

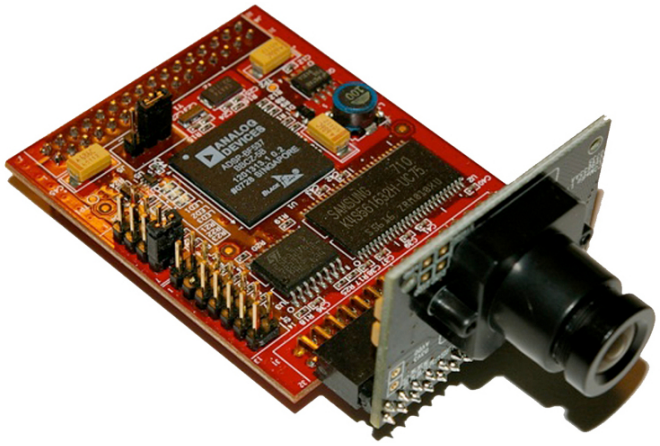
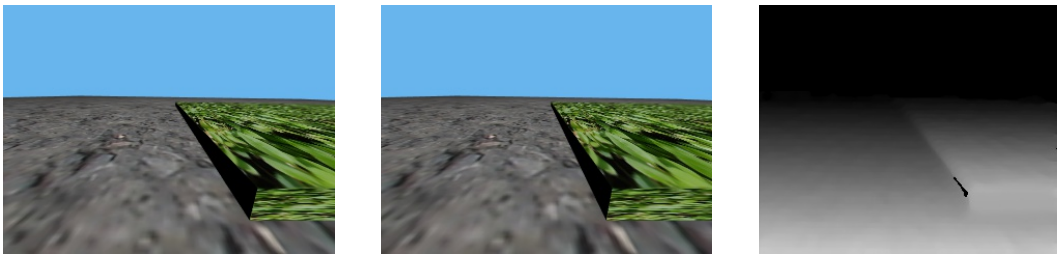


Figure 4.18: SRV-1 Board *(picture from [91]) used to implement the Stereo vision system*

The images from these cameras were calibrated using OpenCV [92] libraries. This calibration procedure calculates the intrinsic and extrinsic parameters of the camera, which will then allow the images to be rectified in order to align the epipolar lines, thus allowing for an easier matching between points in the scenes. These calibrated images were then fed to a disparity calculation system, implemented again through the use of the OpenCV libraries.

The system was then tested in the lab environment and was confirmed to generate correct disparity maps. The original reason to use these cameras was to have the disparity calculation system implemented onboard the cameras, but due to the unavailability of the physical robot, this was left in software and this physical implementation was never integrated with the remaining systems. Instead, a software model was developed using two virtual cameras set up in the robot simulator, placed 6*cm* apart, the same as the planned physical implementation. Their images were then fed to the same disparity calculation module used in the physical setup. An example of the results this approach provides can be seen in figure 4.19.



(a) Left image **(b)** Right Image **(c)** Disparity Map

Figure 4.19: Example of the result obtained from the disparity calculation module *From the left* ***(a)*** *and right* ***(b)*** *images, the module generates a disparity map* ***(c)****. In the disparity map, darker colours mean higher distance.*

4.2.2 Spiking Output

As the vision processing and learning systems were all implemented using spiking signals, a new output layer, formed by spiking neurons, was attached to the disparity map generation system.

This layer consists of disparity sensitive integrate and fire neurons. The inputs for these neurons are the disparity values calculated by the stereo vision system described in the previous section. Each pixel in the disparity map feeds a set of neurons and each of these neurons has a Gaussian shaped response to a particular value of disparity. The neurons attached to each pixel are distributed across a range of disparity values as shown in Figure 4.20 and explained in more detail below. A linear distribution of the disparities across the neurons was chosen as it provides a higher depth resolution for closer objects, since for these, the same difference in disparity represents a smaller difference in depth than for objects located further away (figure 4.21).

The generic Gaussian shaped function is represented by:

$$f(x) = ae^{-\frac{(x-b)^2}{2c^2}}, \tag{4.28}$$

where a is the height of the curve, b is the position of its peak and c controls its width. In order to simplify and speed up the computations required, a was set to 1. Parameters b and c were calculated dynamically according to the range required and number of neurons available.

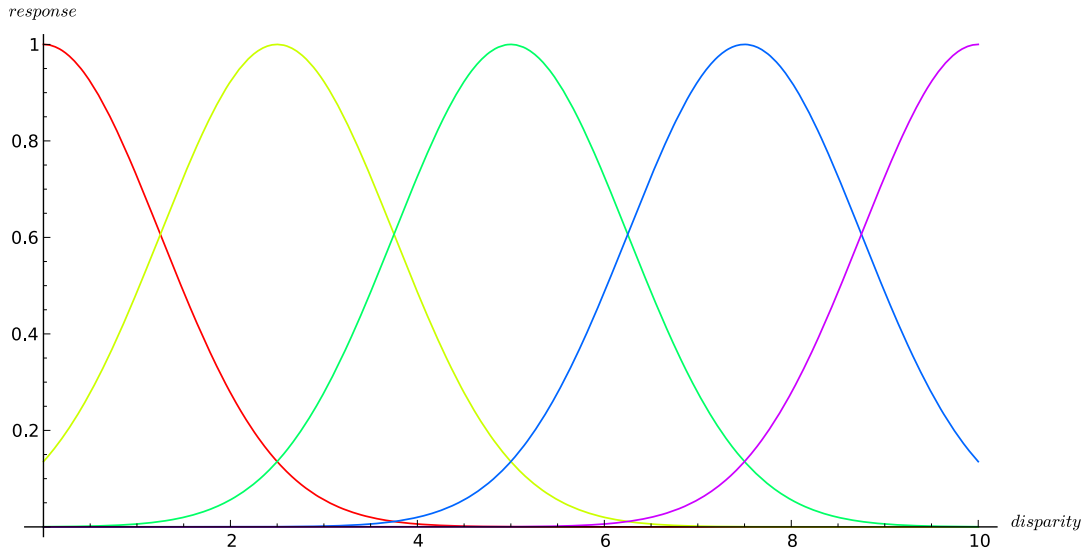


Figure 4.20: Distribution of the disparity sensitivity throughout the different neurons. *The Gaussian shaped response curves in the neurons were tuned so that there would be an overlap between them.*

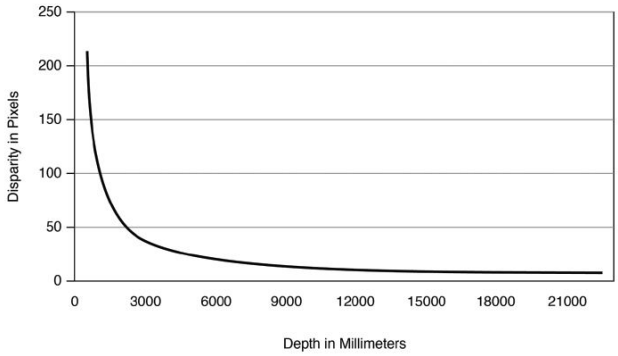


Figure 4.21: Relationship between disparity and depth *(plot from [93]) "Disparity values as a function of depth, assuming a focal length of 8 mm, baseline of 10 cm, and pixel size of 7.5 microns."*

The centres of the curves were evenly distributed between the minimum and the maximum disparities, therefore the difference (d) between the curve peak in consecutive neurons is given by:

$$d = \frac{maxDisparity - minDisparity}{N_{neur} - 1}, \tag{4.29}$$

where N_{neur} represents the number of disparity sensitive neurons attached to the disparity map pixel, $maxDisparity$ and $minDisparity$ are the positions of the curve peaks with the highest and lowest disparity centres respectively. With this, the final position of each neuron $i = 0, \ldots, N_{neur}$ can be defined by

$$b_i = i * d + minDepth \tag{4.30}$$

In order to allow some overlap between the response curves in successive neurons, c is set to $\frac{d}{2}$.

An example of the output of this layer can be seen in figure 4.22.

The pseudocode for the disparity sensitive neuron can be seen in algorithm 4.1.

Algorithm 4.1 Disparity Sensitive Neurons

```
function GET_OUTPUT(input)
    membraneVoltage ← membraneVoltage + e^(−(input−b)^2 / 2c^2)
    if membraneVoltage ≥ threshold then
        membraneVoltage ← 0
        output ← 1
    else
        membraneVoltage ← membraneVoltage * 0.99
        output ← 0
    end if
    return output
end function
```

These neurons signals are then routed to the stabilisation module described in Section 4.3, the output of which consists of spikes reacting to approaching objects.

4.3 Obstacle Detection/Approach detection

For the vision processing area, a biological inspired approach was also used. In several animal species, neurons situated in a particular cortical area are sensitive to

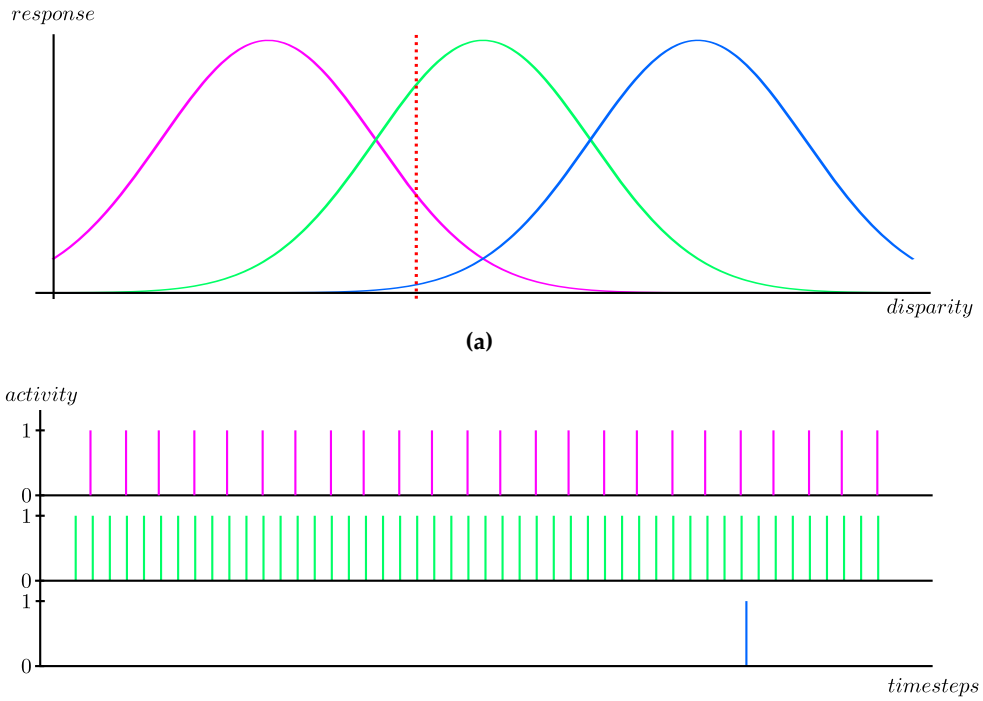


Figure 4.22: Example for the disparity calculation output. ***a*** *shows the sensitivity curves for 3 neurons (purple, green and blue) and the disparity value which is being fed to the neurons (red doted line).* ***b*** *shows the output of the 3 neurons given the input disparity. The green neuron is the most sensitive to the input value, therefore it has the highest spiking frequency of the three. The purple neuron has a lower sensitivity therefore it is still activated, only with a smaller strength, thus having a lower spiking frequency. The blue neuron barely has any sensitivity to the disparity value, so it almost does not fire.*

different levels of disparity [94]. This served as inspiration for this system.

As already explained in section 4.1, the main problem related to legged locomotion as opposed to wheeled locomotion is an increase in body movement. This movement introduces some difficulties for the vision processing, especially its angular component.

Due to the amount of problems this movement causes in the depth from motion algorithm, a stereo system was implemented instead. Because this type of system does not rely on dynamic characteristics of the movement, but is instead time independent, the depth is calculated correctly, but the movement is transferred to the next system. In this case, the object approach detection system, whose purpose is to provide the predictive path of the learning system with events alerting when the distance between the robot and objects in the scene is decreasing.

The angular movement caused by the walking inserts noise into this detection, as the up and down movement generate an apparent cyclical approach and distancing of the scene to the camera, which causes alert events to be erroneously generated. This will in turn cause abnormal correlations in the learning system, therefore degrading its performance.

In order to counteract this problem, and generate the correct events, a spiking network with filtering capabilities was created for each of the disparity map pixels.

4.3.1 Layer Connections

Each disparity sensitive neuron is connected to a habituation neuron by an excitatory synapse with a weight of 0.5. There are also inhibitory synapses connecting each disparity neuron with all the habituation neurons fed by lower disparity neurons with a weight of 1.0 (the signal goes through unchanged). These inhibitory connections will cause a temporary suppression of the higher disparity neurons which will cancel any high frequency variation between levels, thus behaving slightly like a low pass filter with a bias to the higher disparities. This bias does not have a negative impact, as the learning algorithm will adjust to this. This architecture is illustrated in figure 4.23.

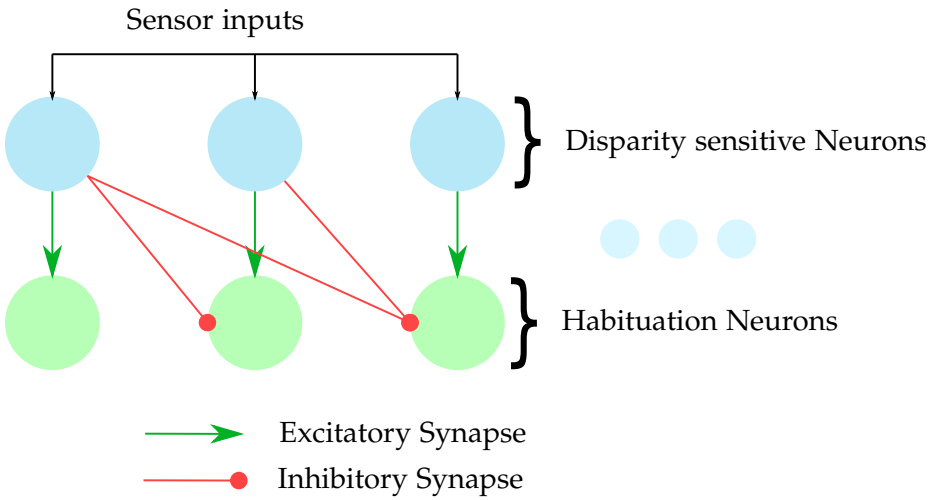


Figure 4.23: Architecture of the obstacle detection system. *The output layer of the disparity calculation system is connected to the habituation neurons on a one to one basis through excitatory connections and they also have inhibitory connections to the habituation neurons of lower disparities (i.e. higher distances).*

4.3.2 Output Layer

The output layer consists of integrate and fire neurons which implement a habituation behaviour. This habituation behaviour will allow the system to detect novelty instead of general activity. When a certain disparity neuron becomes active, its companion habituation neuron will start firing at the same rhythm but as the activity continues and the novelty factor wears off, the output neuron will become less and less active due to habituation. This network will therefore provide a detector of approaching objects that returns the disparity at that image region.

For the initial testing the oscillation movement induced by the walking was modelled by a sine wave with a frequency of $0.5Hz$. This value was chosen as it approximately represents a full walking cycle at normal speed. In this demonstration 6 input columns were used, the minimum distance was set to 7 and the maximum distance to 23.

When the sensor readings from the walking model were fed into the input (disparity sensitive) neurons the activity changed from neuron to neuron according to which neuron was more sensitive to the particular disparity. This is shown in figure 4.25.

Algorithm 4.2 Habituation Neuron

function GET_OUTPUT($excitatoryInput$, $inhibitoryInput$)
$membraneVoltage \leftarrow membraneVoltage + ((1 - habituationStatus) * excitatoryInput - inhibitoryInput)$
if $membraneVoltage \geq threshold$ **then**
$membraneVoltage \leftarrow 0$
$habituationStatus \leftarrow min(1.0, habituationStatus + habituationFactor)$
$output \leftarrow 1$
else
$membraneVoltage \leftarrow membraneVoltage * 0.99$
$habituationStatus \leftarrow max(0.0, habituationStatus - habituationDecay)$
$output \leftarrow 0$
end if
return $output$
end function

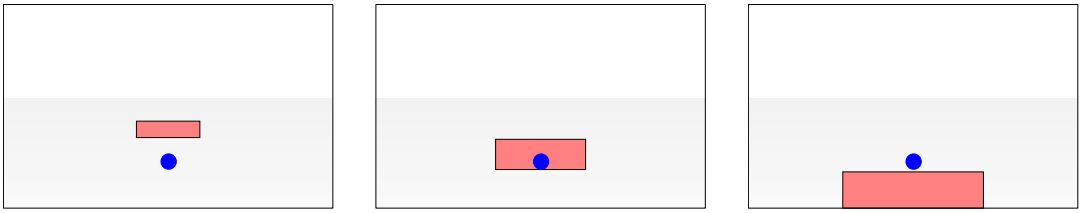


Figure 4.24: Three phases for a simple object approach. *In* **(a)** *the camera is approaching the obstacle but the pixel being analysed is still covering the ground only. In* **(b)** *the pixel starts to cover the object, so the disparity value of that pixel increases, as the distance decreases. In* **(c)** *the obstacle moves out of the pixel coverage, so the disparity value jumps back to the one corresponding to the ground.*

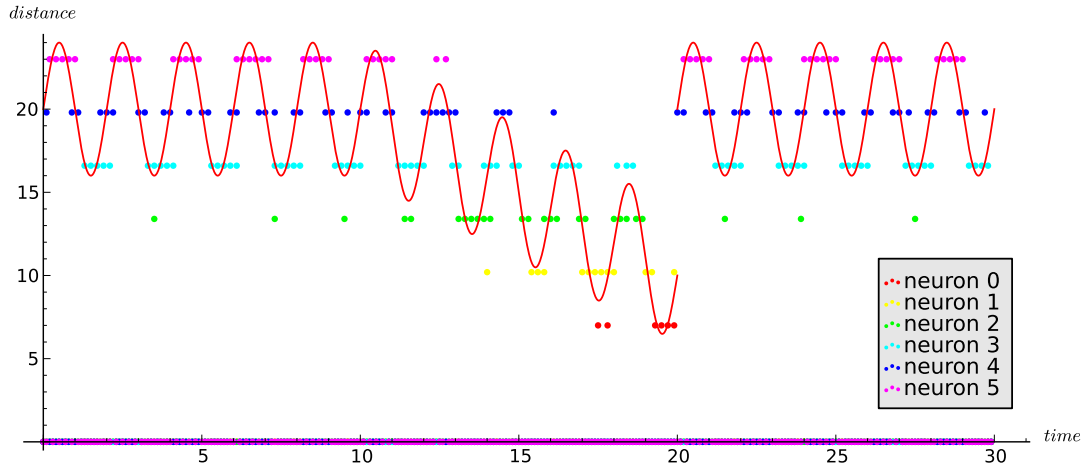


Figure 4.25: Output from the disparity sensitive neurons when stimulated with a sinusoidal curve. *The curve in red represents the distance measured for a generic point in the image. The first stage represents a situation on which there are no obstacles present in the pixel, therefore the curve represents only the distance from the camera to the ground with a sinusoidal component which represents the change in values caused by the movement of the robot. In the second phase, the pixel starts to cover the object and the distance measured by the pixel starts to decrease. In the third phase the object leaves the pixel coverage and the distance value goes back to the one corresponding to the ground. These phases are illustrated in figure 4.24*

Figure 4.26 shows the output neurons activity when using the same inputs as for figure 4.25, it can be seen that after a period of adaptation, the neurons start detecting the ground but quickly start to slow down as the habituation begins to increase. When the distance starts decreasing, this is correctly detected by the corresponding neurons and when the distance returns to the ground level it causes the habituation to activate again, thus keeping the activity low.

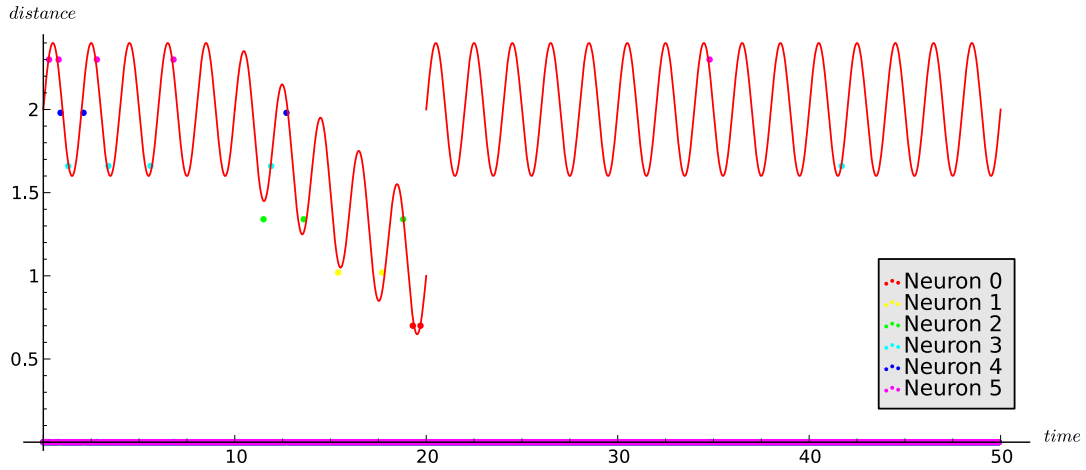


***Figure 4.26: Output neurons activity**. Neurons 3 to 5 start to fire in response to the ground distance, but their activity quickly slows down. When the object starts to get closer, neurons 2, 1 and then 0 successfully detect its approach. After the object passed the pixel being analysed, the disparity measured goes back to the ground level, but due to the habituation effect, the activity in neurons 3 to 5 stays very low.*

4.4 Summary

In this chapter, the shortcomings of the depth from motion algorithm were explained. These were mainly caused by two factors: its overzealous noise cancelling system which, when coupled with a slightly noisy camera, discards most of the calculated depths; and the assumption that the optical flow field is perfectly radial, which does not apply in the case of this legged robot as the movement caused by the locomotion will add translational and rotational components to the optical flow.

In order to replace the depth from motion system, the object approach detection system developed made use of a stereo vision setup as its source of depth data. This system used biologically inspired components and mechanisms, such as integrate and fire neurons, excitatory and inhibitory connections, disparity selectivity and habituation to achieve its function. In the standalone simulations, the system successfully detected incoming obstacles, so, in the next phase, the system was integrated with the rest of the models for a full simulation of the robot. This will be described in detail in Chapter 5.

Chapter 5
Robot Model Integration

As already mentioned, the main motivation of the project was to test the hypothesis that it is possible to effectively control a legged robot by integrating a set of biologically plausible systems.

The project consisted of three main packages. The physical robot with its corresponding control system, the vision system and the learning system (the connections between the several components can be seen in figure 5.1).

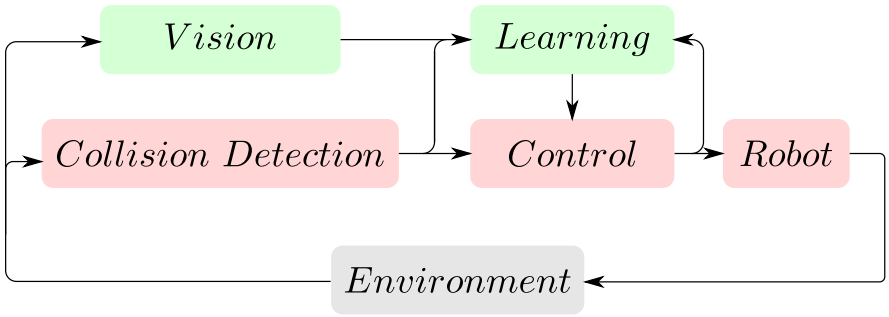


Figure 5.1: Connections between the system components. *Before learning takes place, the main systems are the ones in light red boxes. The control system generates the movements required for the robot to walk through the environment. When the robot collides with an obstacle, this is detected by an increase in the motor joint load which then sends a signal to the control system to generate a reflex action. As this event happens, the learning system learns the correlation between the vision system output and the occurrence of collisions and also the actions taken in those circumstances. After learning takes place, the learning block will start to modulate the control system activity, based on the vision system, in order to avoid collisions.*

The integration of these systems was done in several iterations, with increasing complexity, in order to allow for an early detection of potential problems.

In the first iteration, described in section 5.1, the learning system was tested by connecting it to a simple control system which was responsible for the movement of a simulated foot through an artificial 2D world with step like obstacles. In this system, instead of a vision system, a very simple prediction signal was used.

The next step was to integrate the vision system with the learning system within a more complex 3D simulator and, following that, another integration step was taken by closing the loop and having the learning system modulating the robot gaits. Both of these integration steps are discussed in more detail in section 5.2.

After a brief summary describing the packages involved, the experiments performed will be described and analysed.

Vision system

As discussed in Chapter 4, the vision system originally planned to be used, was based on a “depth from motion" algorithm but, due to the limitations discussed in section 4.1, a system based on stereo vision was implemented instead (sections 4.2 and 4.3). This system uses a standard disparity calculation system to feed a spiking neural network which performs filtering and detects approaching objects.

Learning module

As explained in Chapter 3, the inputs for this module are one or more predictive pathways and one reflexive pathway. In these simulations, the vision system provides the predictive pathways, whilst the reflexive pathway is provided by the load trigger implemented in the joints.

Each of the predictive pathways receives its input from an output neuron of the obstacle detection system. Each of these is responsible for signalling the approach of an object at a certain distance at specific positions in the field of view (each detection block is responsible for detecting an approaching obstacle at a certain voxel in the visible area in front of the cameras). The learning system then correlates these inputs with the reflexive input and updates the weights for each of the connections accordingly.

In this system, the proportionality of the outputs to the height is accomplished thanks to the use of parallel pathways for the obstacle detection, each responsible for a sections of the field of view. Since the number of active obstacle detection blocks will be

proportional to the occupancy of the said object in the field of view, the output of the prediction system will be proportional to the height of the object.

5.1 Closed Loop Proof of Concept

As soon as the behaviour of the learning algorithm implementation was believed to be correct, a very simple software simulation was set up using Python. This simulation was developed in order to try and emulate a control system responsible for the movement of a single leg. This simplification was performed in order to trial the learning concepts in a simpler and easier to analyse framework where the physical setup of the leg is abstracted to the position of the foot on the vertical plane parallel to the direction of the robot movement.

The controller implemented normal walking as well as two different reflex behaviours: the elevator and the search reflexes, illustrated in figure 5.2. Both of these are present in actual insects, which served as inspiration to the robot [95, 96].

Algorithm 5.1 Normal Walking Generation

$movementCentre_x \leftarrow foot_x + movementHorizontalRadius$
$movementCentre_y \leftarrow foot_y$
$angle \leftarrow 0$
while $foot\ does\ not\ collide\ with\ surface$ **do**
  $angle \leftarrow angle + angleStep$
  $x_{foot} \leftarrow movementCentre_x - movementHorizontalRadius * cos(angle) + searchInput - climbInput$
  $y_{foot} \leftarrow movementCentre_y + movementVerticalRadius * sin(angle) + climbInput$
end while

The **search reflex** (figure 5.2b) is activated when a leg does not find a foothold at the expected position after a nominal step. This reflex has the purpose of finding a foothold further away from the current foot position. It initiates a cycle where it keeps performing more step movements further and further forward until it either finds the other side of the hole and continues walking or reaches the maximum stretch of the leg, in which case the hole is impossible to negotiate. The inner workings of this system can be seen in algorithm 5.2.

The **elevator reflex** (figure 5.2c) is activated when the leg collides with an obstacle while in the swing phase and it consists of successive attempts to reach the top of

Algorithm 5.2 Search Reflex Generation

$movementCentre_x \leftarrow foot_x + movementHorizontalRadius$
$movementCentre_y \leftarrow foot_y$
$angle \leftarrow 0$
while $foot\ does\ not\ collide\ with\ surface$ **do**
$angle \leftarrow angle + angleStep$
$x_{foot} \leftarrow movementCentre_x - movementHorizontalRadius * cos(angle) + searchInput - climbInput$
$y_{foot} \leftarrow movementCentre_y + movementVerticalRadius * sin(angle) + climbInput$
end while

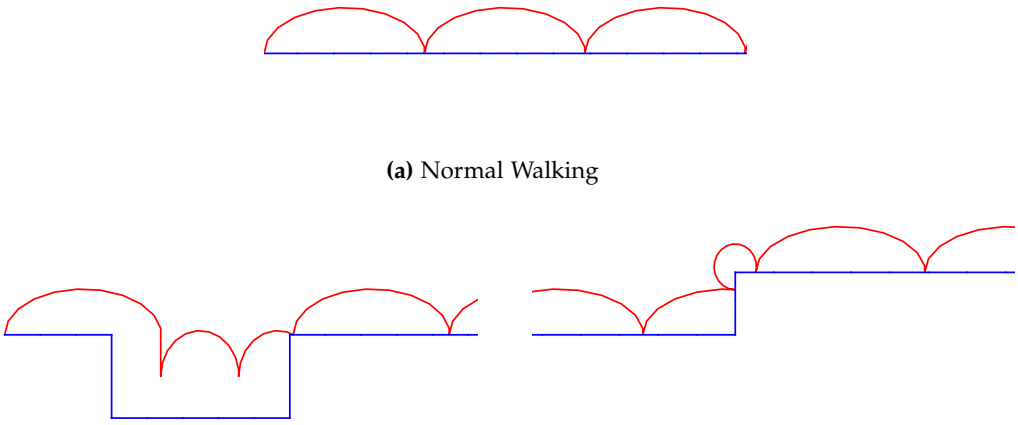
that obstacle. When the collision is detected, the leg retracts slightly from its position and then tries to move forward again, while continuing to move up until it finds the top of the obstacle. The inner workings of this system can be seen in algorithm 5.3.

Algorithm 5.3 Elevator Reflex Generation

$movementCentre_x \leftarrow foot_x + movementHorizontalRadius$
$movementCentre_y \leftarrow foot_y$
$angle \leftarrow 0$
while $foot\ does\ not\ collide\ with\ surface$ **do**
$angle \leftarrow angle + angleStep$
$x_{foot} \leftarrow movementCentre_x - movementHorizontalRadius * cos(angle) + searchInput - climbInput$
$y_{foot} \leftarrow movementCentre_y + movementVerticalRadius * sin(angle) + climbInput$
end while

A surface with several obstacles was then created in order to set a benchmark for the various implementations. A first test was performed with the simple controller in order to have a benchmark for the number of timesteps needed for the foot to go through the whole course. As the reflexes were activated often, it was expected that the total time taken would be greater than the implementations which used learning and will be described later on.

The spiking ISO3 model was then connected to the simulator described above. As previously described, the ISO3 algorithm needs two types of inputs: a reflex signal eliciting an immediate reflex action from the system and a predictive input, the weights of which will be learned. In this case, the reflex event was triggered by the collision between the foot and the walls of the obstacles and the prediction event was issued by the crossing of a threshold in the proximity of the leg to an obstacle.



(a) Normal Walking

(b) Search Reflex

(c) Elevator Reflex

Figure 5.2: Walking Behaviours implemented. *Trace of the foot positions (red) and terrain (blue) exhibiting the three different behaviours implemented. In the search reflex* ***(b)****, when the foot does not find ground at the expected location, it start searching for a foothold on the other side of the gap. In the elevator reflex* ***(c)****, the foot recedes slightly and tries another step at a higher position in order to climb the obstacle.*

After a few tests, it became apparent that the most useful moment to issue the prediction event was at the start of the swing phase. When issued at this point, the predictive signal had the largest impact on the foot trajectory. This effect was already described in [97]. The obstacle detection system was then modified so that the prediction event was issued only at the start of this phase, when there was an obstacle at a distance of less than a step-length.

Some experiments were then performed and these revealed promising results. After learning took place, the simulated foot was capable of overcoming most of the obstacles without activating the reflex signal (figures 5.5 and 5.6). The several stages of the behaviour as learning takes place can be seen for both the steps and the gaps in figures 5.3 and 5.4 respectively.

In some simulation runs, however, there were still some visible problems. One of these problems was caused by the disconnection between the predictive signal and the distance at which an object was located (within the detection interval), which caused the weight increase for the predictive pathways to be identical independently from the distance to the obstacle. The fact that the system did not take the value of the distance to the obstacle into consideration meant that in circumstances where

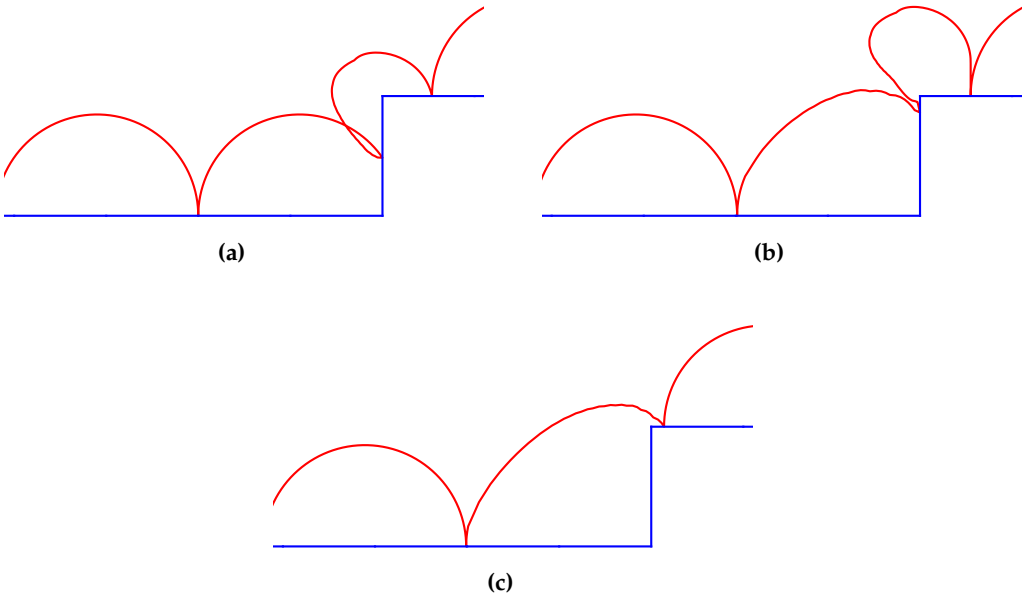


Figure 5.3: Step learning progression. *Traces of the foot positions during a step over an obstacle before* ***(a)****, during* ***(b)*** *and after* ***(c)*** *learning. These were collected during the simulations performed.*

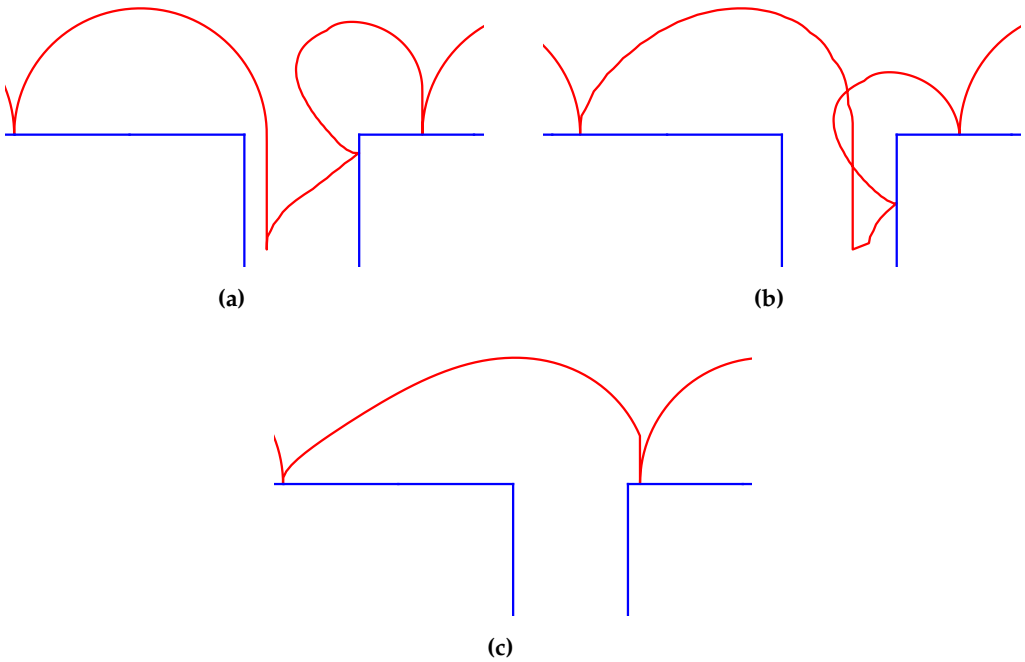


Figure 5.4: Gap learning progression. *Traces of the foot positions during a step over a gap in the ground before* ***(a)****, during* ***(b)*** *and after* ***(c)*** *learning. These were collected during the simulations performed.*

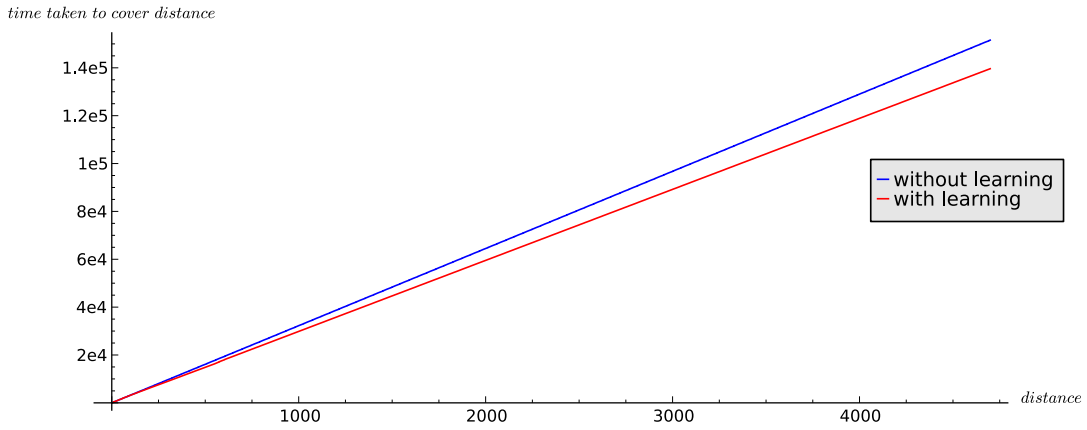


Figure 5.5: Comparison between the simulation runs with and without learning. *As can be seen by the figure, the experiment where learning was activated took less time to travel the same distance. In the beginning of the simulations both lines start with the same inclination, as learning still has not taken place. As soon as learning start to be able to avoid some of the collisions, it avoids the reflex actions which cause the system to travel through the terrain in less time than when learning is not present.*

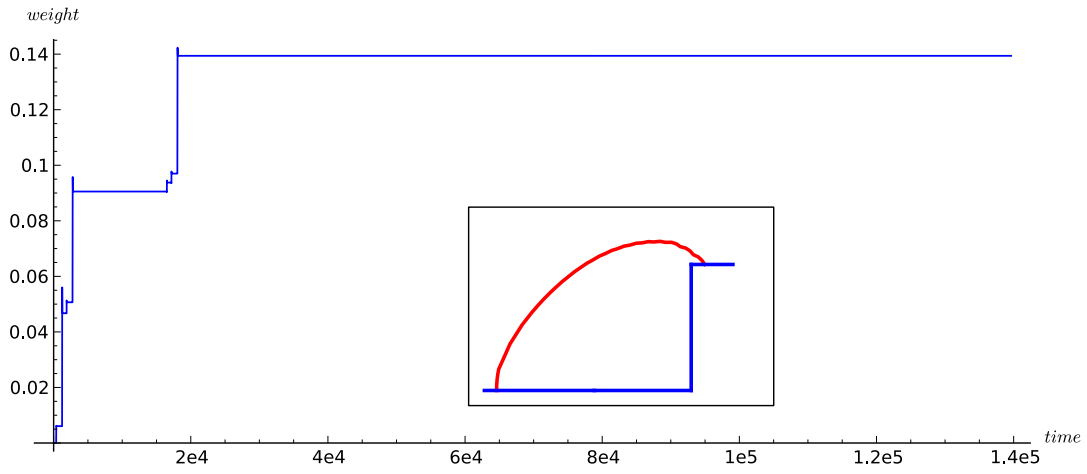


Figure 5.6: Weight progression for the predictive pathway on a successful trial. *Learning takes place mostly at the start of the simulation, after that the foot is able to overcome all obstacles without the activation of further reflex behaviours.*

the foot is very close to the obstacle, it would not be able to rise enough to negotiate it, which then generated a reflex action that caused an increase in the weight, thus increasing the reaction in all other cases as well.

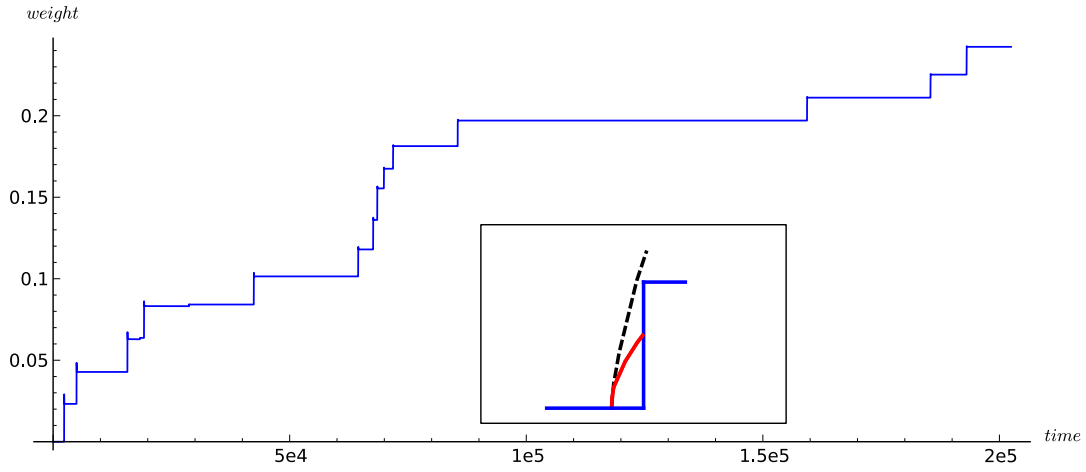


Figure 5.7: Weight progression for the predictive pathway on a trial where some problems were observed. *In this case, learning does not stop as soon as in the previous example, as the terrain had obstacles in tighter intervals that caused the foot to land very close to their wall in certain occasions (as illustrated in the insert, where the dotted line demonstrates the foot trajectory required to overcome the step at that distance), which caused more reflex events to be generated along the duration of the trial.*

Another problem observed was when the system was trained initially with shorter obstacles and then with taller obstacles. As in the previous case, the predictive signal was independent from the size of the obstacles, which meant that after learning it would react in the same way to different sized obstacles. This meant that, after these trials, the weight was always adjusted in a way that optimised the behaviour to respond to the largest obstacles. This caused the foot trajectory to be excessively high on small obstacles, thus waisting time and energy with the longer trajectories. To illustrate this problem, the system which gave the results from figure 5.7 was used for a second run in a terrain with the same characteristics, only with obstacles which were nearly twice the size. As can be seen in figure 5.8, the system had some reflex events, which translated into large increases in the weight. This means that its reactions to any obstacle will now be stronger, independently from its size.

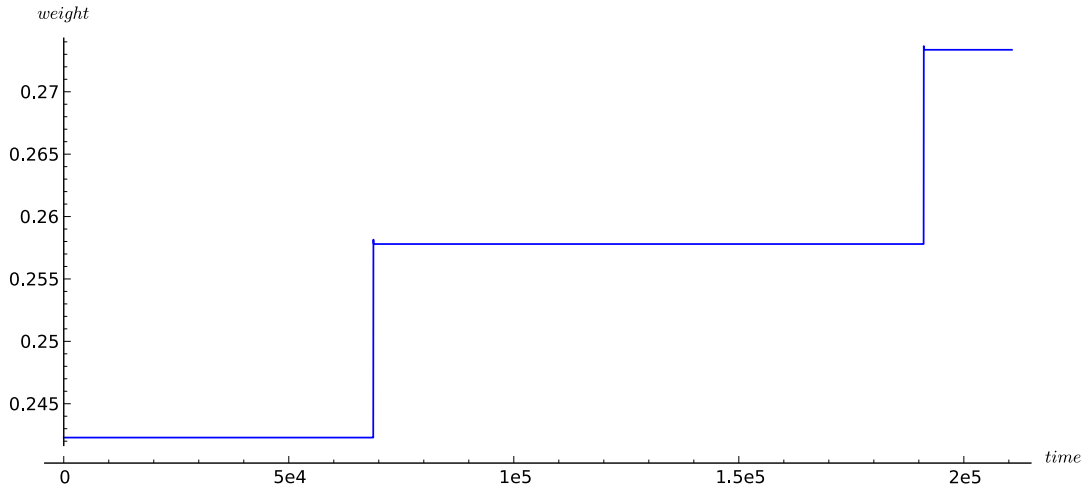


***Figure 5.8: Weight progression for the predictive pathway on the second trial**, which consists of transposing larger obstacles. It can be seen that the weights increase during this trial, as the weights previously learned were not enough to overcome the taller obstacles.*

5.1.1 Proportional Prediction Events

In order to avoid the problems described above, the obstacle detection system was changed. Instead of generating a single spike, the system was setup so that it generates a spike burst with a number of spikes approximately proportional to the apparent height of the obstacle. This apparent height is directly proportional to the height of the obstacle and inversely proportional to its distance. The intuition for this is that the closer or taller the obstacle is, the larger its apparent size and therefore the higher the reaction of the detection system needs to be. After this change was implemented, the system was tested in the same problematic terrain used for figure 5.7 and the system performance was much better. Learning completed successfully early in the trial and all remaining obstacles were overcome successfully without the generation of any further reflex events. This can be seen by comparing figures 5.7 and 5.9. With the non-proportional predictive signal, the system was not capable of achieving weight stabilisation, having been through 16 reflexes with the last one at timestep 193049. With the progressive predictive signal, weight stabilisation was achieved after 8 reflex actions with the last one being at timestep 76686.

The system was then also tested for a second run on the larger obstacle terrain used

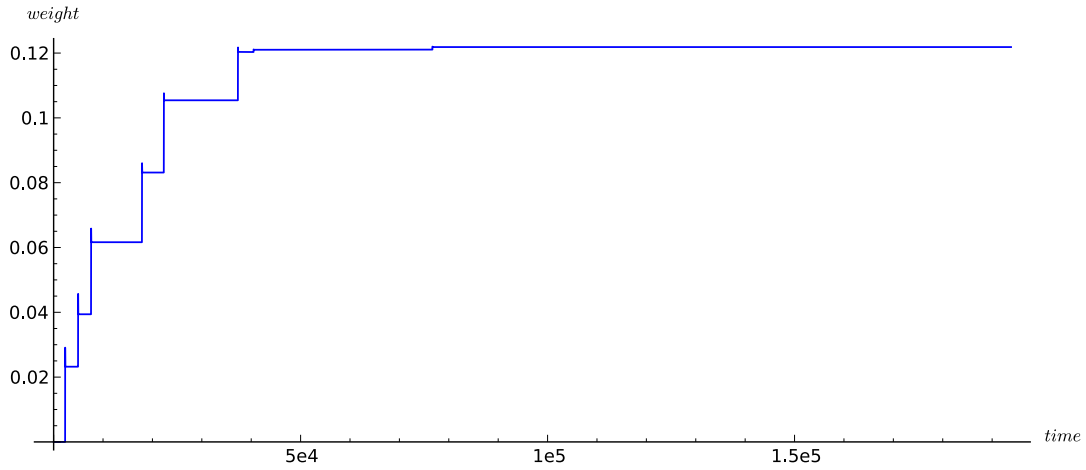


Figure 5.9: Weight progression for the predictive pathway for the system with a proportional predictive signal *using the same terrain as the one used for the trial illustrated by figure 5.7. It can be seen that learning takes place at the beginning of the trial and no more reflex events are generated for the duration of the trial.*

for figure 5.8 and, as expected, its performance improved greatly. The system was able to successfully negotiate the larger obstacles without colliding a single time with any of them, thus maintaining the value of the weight. This was due to the proportional activation generated by the predictive system being enough to compensate for the variations in obstacle size.

5.2 3D Simulations

5.2.1 Physical Robot vs Simulation – some Compromises

The physical robot and its control system, were the responsibility of project collaborators. It was designed to be a scale model of a stick insect (figure 5.10b) and its control system was going to be based on SCASM [38].

Due to complications in the design of the control boards, the physical system was not completed within the project time frame. Although a prototype simulator was provided as a backup plan, it proved inadequate for the needs of the project. Due to these circumstances, the integration was performed by using the commercial simulator Webots® [99]. The robot appearance model (figure 5.10a) was developed from

(a) 3D model of the robot

(b) Stick insect (picture from [98])

Figure 5.10: Robot model and a stick insect, which served as its inspiration.

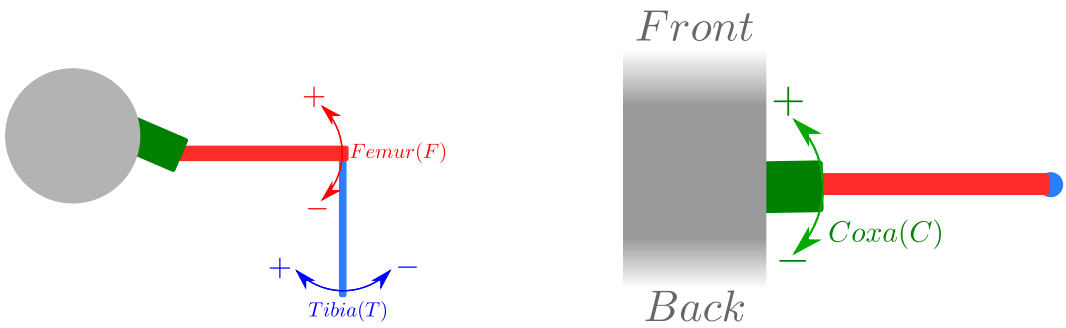
the CAD design of the physical robot. In order to simplify the simulation, the actual physical model and the surfaces used for collision detection were built using simple cylinders and boxes in a way that approximated as much as possible the original shape, rather than the complex triangular meshes used for visualisation. This allowed for real time simulations to be performed while still maintaining a very good level of accuracy in the collision detection of the simulator. The stereo system was also emulated through the use of two virtual cameras in the simulation.

As the control system was part of the same work package, it suffered from the same delay. Because of this, a replacement control system was implemented using a simple CPG, which allowed for the tripod gait to be generated and modulated through various bias inputs. This replacement system also provides load sensing on the front leg joints for the collision detection used in the learning system. These sensors used are analogous to the intended hardware implementation, as the servos used were capable of the same measurements.

5.2.1.1 Basic walking Control System

Each robot leg is composed of three different parts, the coxa, the femur and the tibia. An illustration of these components and of their movements can be seen in figure 5.11.

For the walking control, a simple CPG based system, capable of performing the alternating tripod gait, was used. In this system, all the joints were controlled by



(a) Frontal view **(b)** Top view

Figure 5.11: Robot Leg illustration. *Illustration of the different components of the robot leg and their movements.* ***(a)*** *shows a frontal view and* ***(b)*** *shows a top view. The angles of the coxa, femur and tibia will be represented by C, F and T respectively. In this work, C is positive when the coxa turns towards the front of the robot, F is positive when the femur lifts and T is positive when the tibia rotates towards the robot.*

sinusoidal waves. Depending on the type of joint, the parameters of this wave were different. For the implementation of the alternating tripod gait, two groups of legs were created (figure 5.12.

$$
\begin{aligned}
C_{1L}(n) = C_{2R}(n) = C_{3L}(n) &= -sin(n\frac{\pi}{180}) * 25^\circ \\
C_{1R}(n) = C_{2L}(n) = C_{3R}(n) &= sin(n\frac{\pi}{180}) * 25^\circ \\
F_{temp}(n) &= sin\left(n\frac{\pi}{180} - \frac{\pi}{4}\right) * 10^\circ \\
F_{1L}(n) = F_{2R}(n) = F_{3L}(n) &= F_{temp}(n) * \theta\left(F_{temp}(n)\right) \\
F_{1R}(n) = F_{2L}(n) = F_{3R}(n) &= -F_{temp}(n) * \theta\left(-F_{temp}(n)\right)
\end{aligned}
\tag{5.1}
$$

These two groups reacted symmetrically to the sinusoidal wave, and the exact characteristics of their movement is described by equation 5.1. These signals can be seen in figure 5.13. The signal responsible for the control of the Tibia joint angle is not shown, but it has twice the frequency of the signal controlling the Coxa joint angle with varying time offsets, depending the leg.

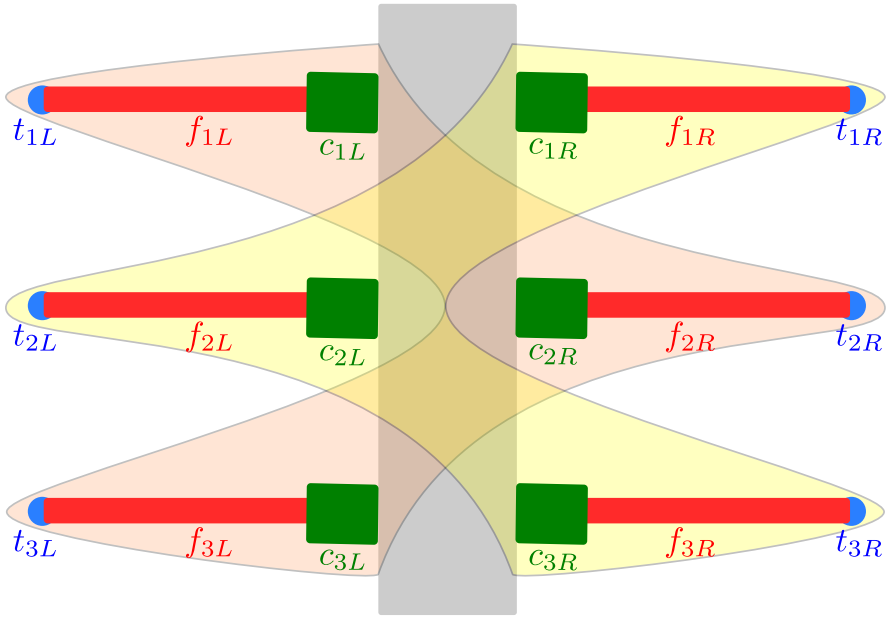


Figure 5.12: Robot leg groups. *Simplified diagram illustrating the two groups of legs used for the tripod gate walking.* 1*L*, 2*R and* 3*L (light red) form one group of legs and* 1*R*, 2*L and* 3*R (light yellow) form the other group.

5.2.1.2 Reflex Implementation

In the robot, the reflex signal is sent through two different pathways, one connected to the Coxa and the other to the Femur. This signal will affect the CPG signal controlling the joints by stimulating a backward motion in the Coxa joint and stimulating the lift of the Femur joint, as seen in figure 5.13.

Due to time constraints, no search reflex was implemented, therefore the experiments performed only involved step like obstacles and no gaps were added to the terrain.

5.2.2 Vision system and learning system integration

In a first phase, the object approach detection system was tested in connection with the stereo system. For this test, the stereo system and the obstacle approach detection system were connected together and the latter was connected to the learning system (figure 5.14). The learning system did not, however, modulate the walking system in this test. The robot was then set to walk towards a step. The goal for this test was to guarantee that the input signals were being correctly generated before performing longer simulations with the full setup.

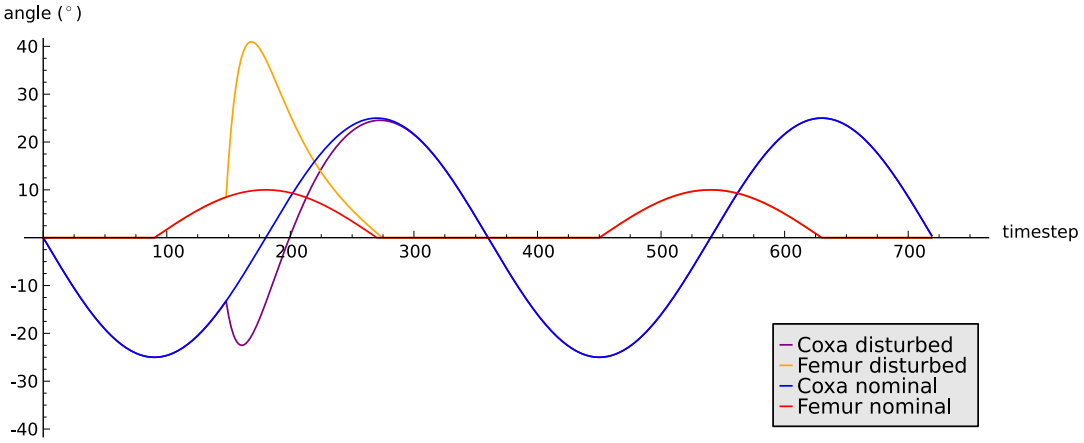


Figure 5.13: Comparison between the nominal and disturbed joint angles of the Coxa and Femur joint angles. *During nominal walking, the coxa angle follows a full sinusoidal wave.The femur is the positive section of a sinusoidal wave with the same frequency, with a phase 90 degrees ahead of the coxa. When a reflex signal is fired, at timestep 150, its trace is added to the different joint activation signals, with different gain parameters, which causes the coxa to recede temporarily and the femur to lift higher.*

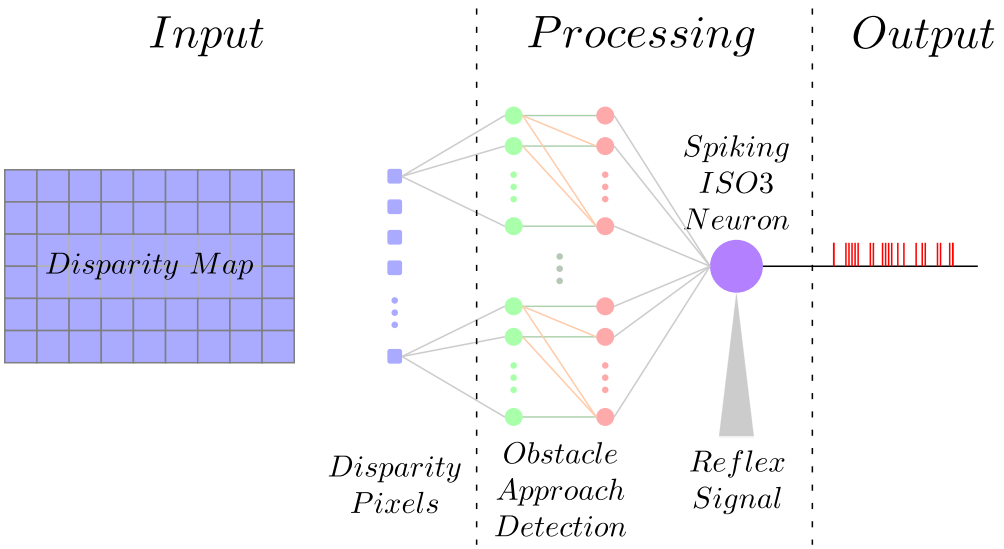


Figure 5.14: Illustration of the integration between the vision and learning systems. *Each pixel of the disparity map calculated with the stereo system is connected to an obstacle approach block described in section 4.3. The outputs of these blocks are the predictive inputs to the Iso3 Neuron. The reflex signal is generated from the load sensors in the front legs.*

The behaviour of the obstacle detection blocks was logged and verified to be correct. The traces for one of the predictive inputs from one of the 3D simulations can be seen in Figure 5.15.

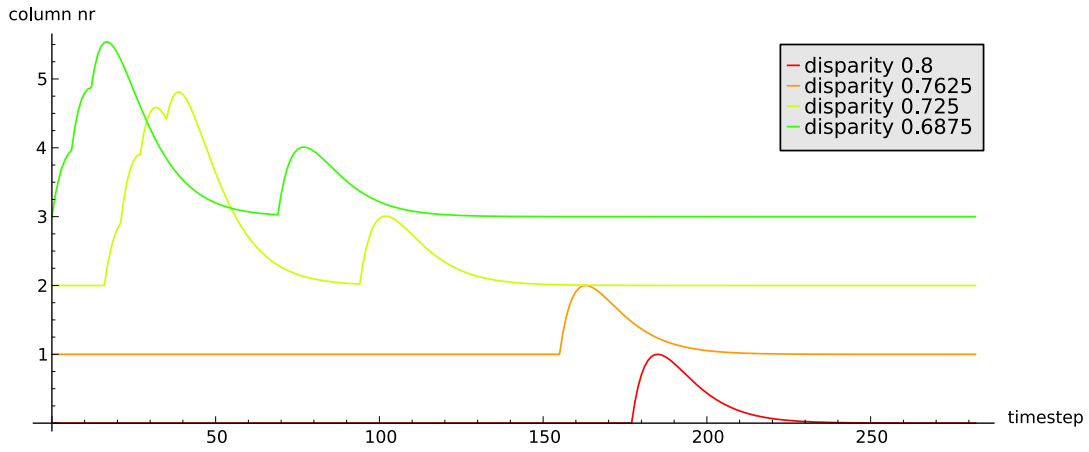


Figure 5.15: Activity in some predictive inputs after passing through the input filter. *Each input was being fed by a different column of one detection block. At the beginning of the simulation some detection columns are very active. This is due to the fact that the habituation process takes a little time to stabilise the neurons. After it stabilises, the activity is mostly limited to new events.*

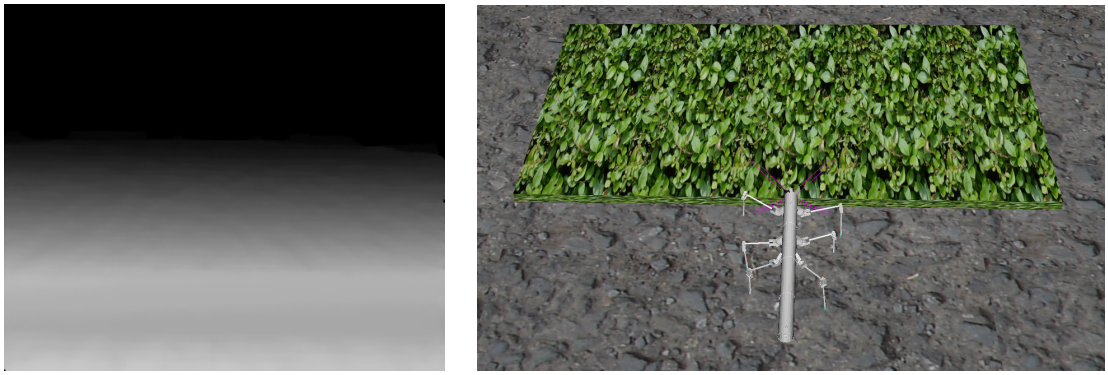
5.2.3 Closed Loop Experiments using Spiking ISO3 learning

After confirming the inputs from the prediction system were working correctly, the final integration was made. The stereo vision and the obstacle detection system were then connected to the Spiking ISO3 learning system, which was also connected to the modulation inputs in the control system. Two learning blocks were used, one for each front leg. In this chapter, the results shown will be for the right leg, as the ones for the left leg are approximately the same and would not add any extra information.

Several experiments were setup in order to examine the effects of the learning system in the behaviour of the robot. These experiments involve placing the robot in front of a box shaped obstacle and activating the system. The robot then walks forward until it reaches the obstacle at different points, according to the experiment. Each time the robot reaches the obstacle without predicting it, it will generate a reflex response, at which point the robot is moved to the initial position again whilst keeping the status of the learning system the same and this cycle is repeated for several iterations. Each

iteration also has a maximum time duration, corresponding to slightly over the time that the robot takes until it reaches the obstacle. This was done in order to end the iterations in which the reflex action was correctly avoided.

Scenario 1 consisted of the robot approaching the centre of the obstacle. When the robot is close to the obstacle (figure 5.16), the obstacle appears as a step in the ground, as it occupies the entire width of the field of view.



(a) Depth map **(b)** Robot position

Figure 5.16: Scenario 1. *Depth map and simulator screenshot from a moment before the collision with the centre of an obstacle.*

After the first test was completed, the log files were analysed and they showed that after a few iterations, the system no longer generated reflex actions and that the right foot was walking over the obstacle instead of against it. The trace from the right foot whilst successfully stepping on to the obstacle can be seen in figure 5.17, whilst the figure showing weight stabilisation can be seen in figure 5.18.

After learning, some areas of the image have higher weights associated than the others. This means that these areas were considered by the system to be more important in the prediction of the collision with the obstacle than the others. For this first example, the cumulative weight plot for the several image areas can be seen in figure 5.19. It can be seen that most of the lower area of the image was considered to be important for the obstacle detection, due to it being activated before all the collisions.

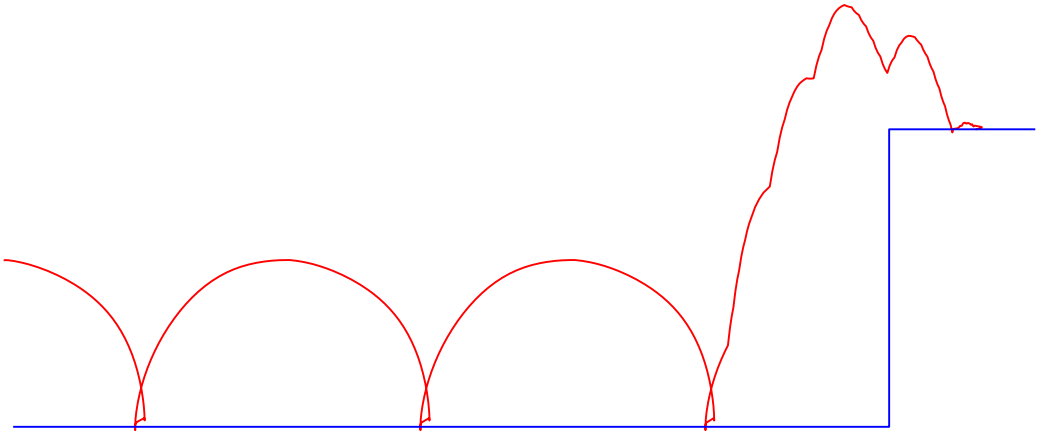


Figure 5.17: Trace of the right foot after learning is successful in overcoming Scenario 1.

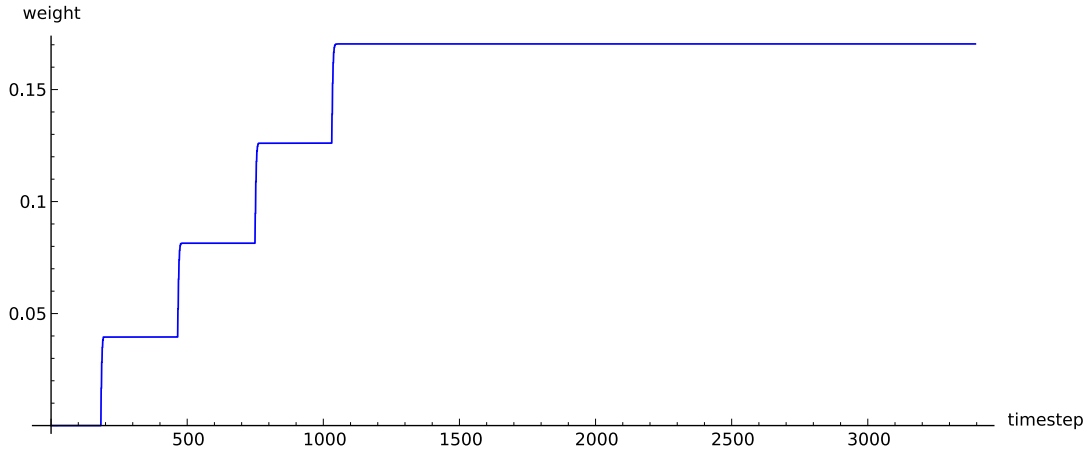


Figure 5.18: Trace of the weights for Scenario 1. *This shows that after a few reflex actions, the weights stabilised due to the disappearance of the reflex action due to successful learning.*

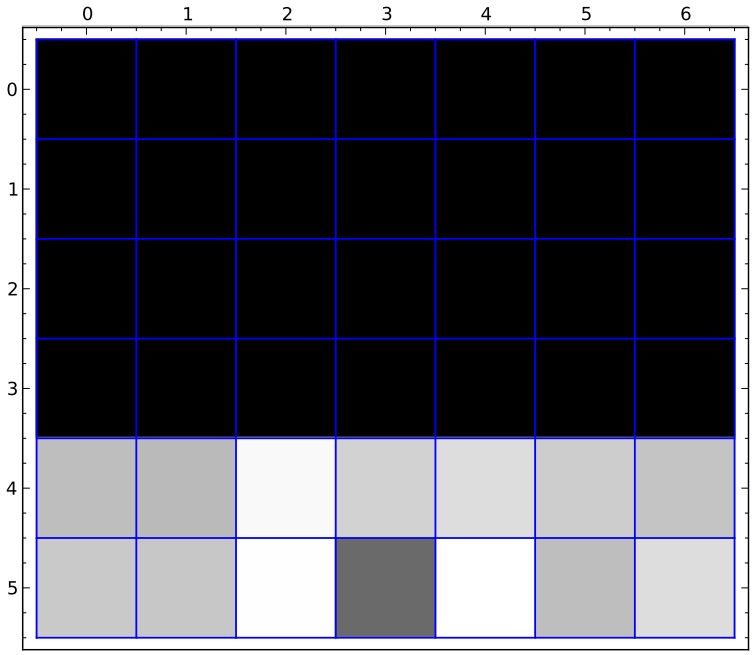
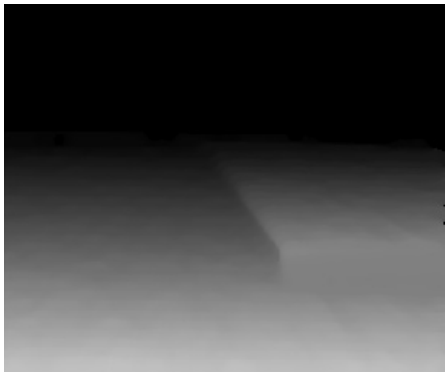


Figure 5.19: Plot of the sum of the weights for each of the blocks at the end of Scenario 1. *This shows the areas in the image which were considered to be more important to predict the obstacle. Black represents 0 weight and white represents the higher of the weights.*

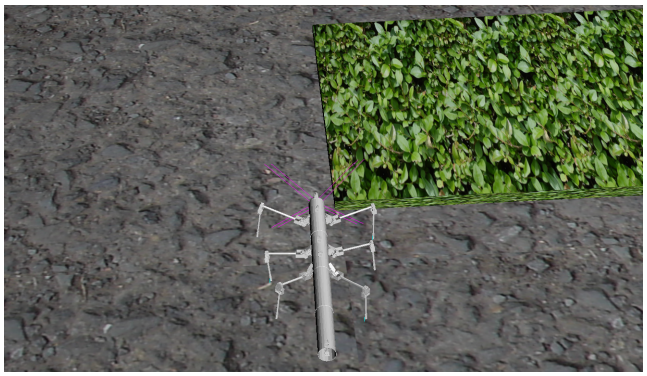
Scenario 2 consisted in placing the obstacle to the right of the robot path (figure 5.20), therefore exposing only the right leg to a collision with it.

Again, the system was capable of learning how to step onto the obstacle (figure 5.21), but this time the area of interest was only the lower right section of the image, as the obstacle that caused the collisions was only detected in that area, therefore all the learning correlations occurred there (figure 5.23). The total weight trace for this experiment can be seen in figure 5.22.

The results of the previous experiments were promising and showed that the system was capable of adjusting the behaviour of the robot in order to avoid complicated reflex actions. The next step was to start mixing the two types of scenarios together.



(a) Depth map



(b) Robot position

Figure 5.20: Scenario 2. *Depth map and simulator screenshot from a moment before the collision with the left edge of an obstacle. This approach will only affect the left leg.*

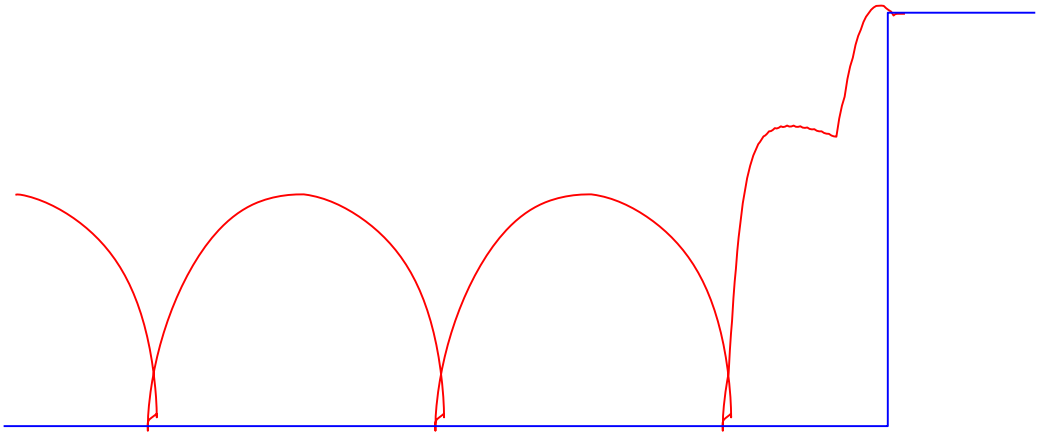


Figure 5.21: Trace of the right foot after learning is successful in overcoming Scenario 2 *Again, like in Scenario 1, the system achieves weight stability, the only difference being that in this case it took more trials until this was achieved due to the smaller amount of inputs correlated with the reflex action.*

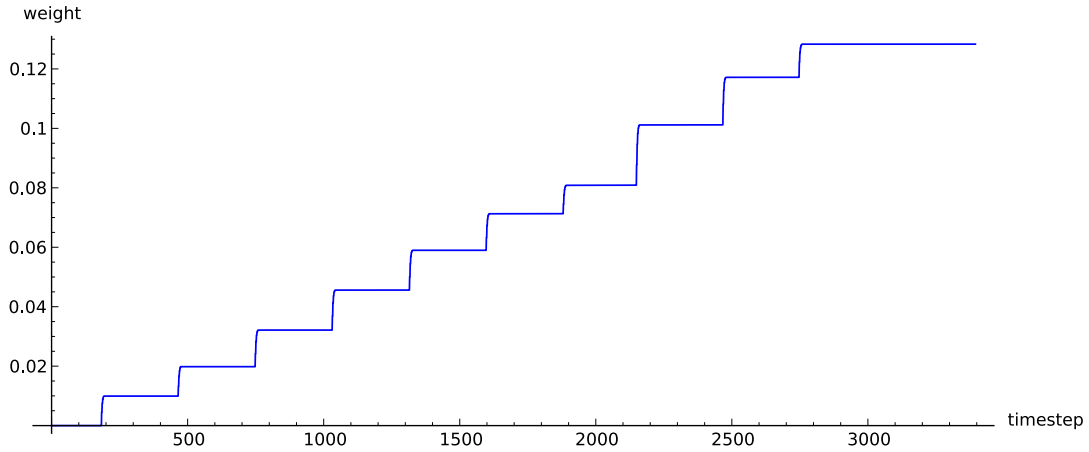


Figure 5.22: Trace of the weights for Scenario 2. *In this experiment the weights took longer to stabilise due to smaller inputs, but eventually stabilised.*

Scenario 3 consisted in having 3 different phases. The first phase was similar to scenario 1 (where the obstacle occupied the entire lower area of the field of view). The second phase was similar to scenario 2 (the obstacle occupied only the lower right section of the field of view), with the exception that instead of starting fresh, it started with the weights learned during the first phase. In the third phase, the object was placed in the left side of the robot path in a way that did not have an impact on the right leg (figure 5.24). The expected action for the robot would be for it to learn how to step over the obstacle on the first two phases but not to react when moving towards the right edge of the obstacle, as this would not generate a reflex for the right leg.

After testing, it was verified that the first two phases had approximately the same result as when it was tested under the individual scenarios, but it did not behave as was predicted when approaching the right edge of the obstacle. In this case, the leg reacted unnecessarily to the obstacle, even though it was in a position which would not disturb the leg movement (figure 5.25)

After looking at the resulting weights for the image areas, it was found that by the end of the second phase, the entire lower part of the image still had high weights (figure 5.26), due to the fact that the learning system had no mechanism which allowed for the decrease of the weights. The only effect of the second phase of this experiment was to increase the weights of the lower right area of the image, but not to change

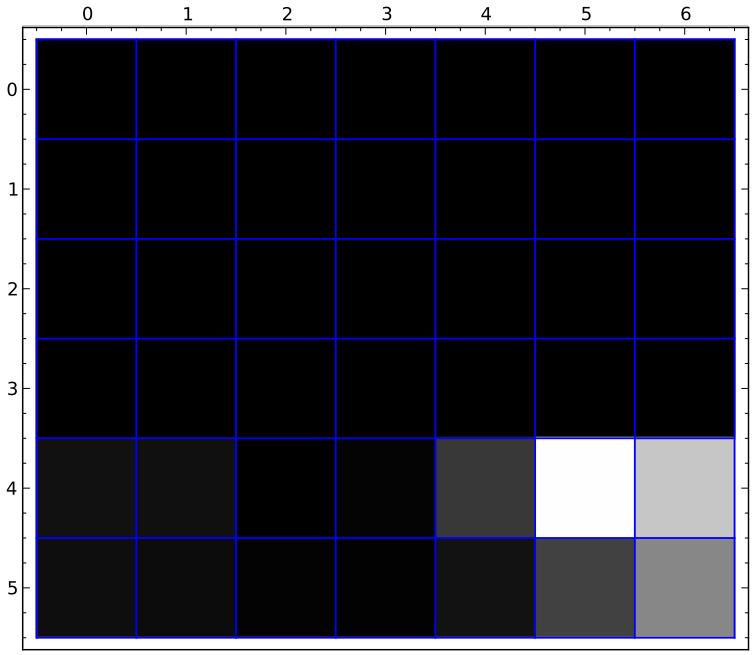


Figure 5.23: Plot of the sum of the weights for each of the blocks at the end of Scenario 2. *This shows the areas in the image which were considered to be more important to predict the obstacle. Black represents 0 weight and white represents the higher of the weights.*

anything else (figure 5.27).

In the previous experiment, there was a split between the 2 phases of learning, first the centre approach and only then with the edge of the obstacle. As this causes most of the weight to be gained during the initial phase, a fourth scenario was devised.

Scenario 4 consisted in two phases, **phase 1** was a merge of the two first phases of Scenario 3 by alternating the position of the obstacle (i.e. middle, right, middle...), while **phase 2** was the same as the last phase of Scenario 3 (i.e. placing the obstacle on the left of the robot so it does not interfere with the right leg). This scenario was expected to help decrease the reliance of the system on the lower left area of the screen by training it earlier on with the different positions. After analysis of the foot trace when approaching the right edge of the obstacle, it was verified that, although slightly better, it still displayed erroneous behaviour (figure 5.28).

Analysing the weights shown in figures 5.29 and 5.30, it can be seen that the problem is still present. Even though there is a small increase in difference between the lower

(a) Depth map **(b)** Robot position

Figure 5.24: Third phase of Scenario 3. *Depth map and simulator screenshot from a moment before the collision with the right edge of the obstacle. This type of approach does not affect the right leg and, as such, should not elicit a response from it.*

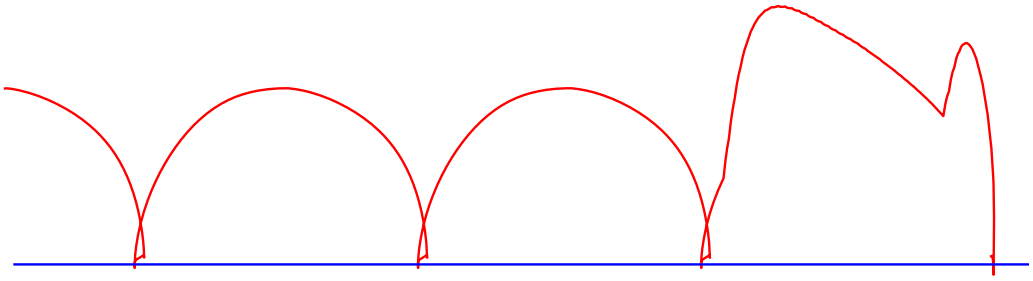


Figure 5.25: Trace of the right foot during one of the iterations of phase 3 of Scenario 3. *The leg reacts to the obstacle, even though it was not necessary to do so.*

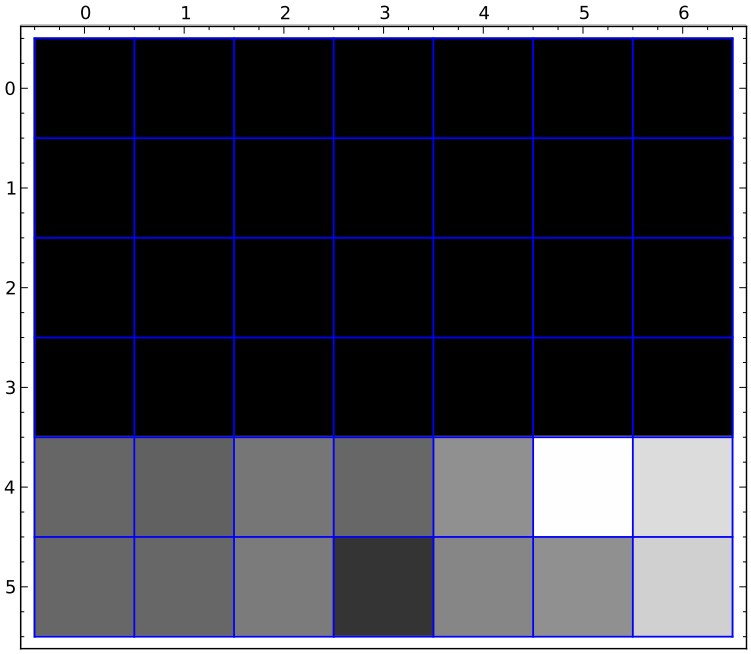


Figure 5.26: Plot of the sum of the weights for each of the blocks at the end of phase 2 of Scenario 3. *In this case, the areas to the right of the image gain importance to a level beyond the rest, but the rest of the areas learned during the training with the wide obstacle still maintain a high weight. Black represents 0 weight and white represents the higher of the weights.*

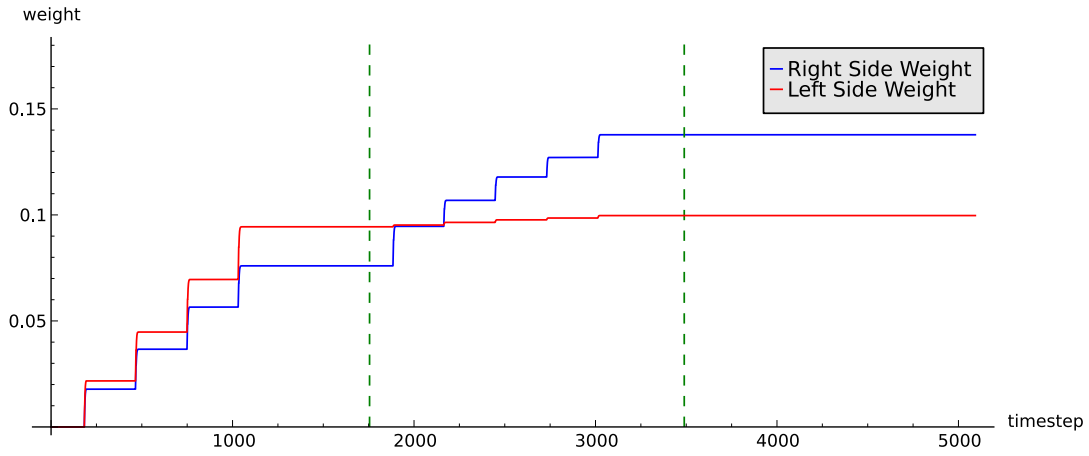


Figure 5.27: Trace of the weights for Scenario 3 for the two sides of the image. *The left side weights represents the sum of weight in columns 0 to 4 of the image, while the right side weights represents columns 4 to 6. In the first phase, both sides of the image increase in weights, in the second phase, a further increase in the weights of the right side can be seen whilst the left side mostly maintains the same weight as in the first phase. In the third phase it can be seen that all weights stay constant.*

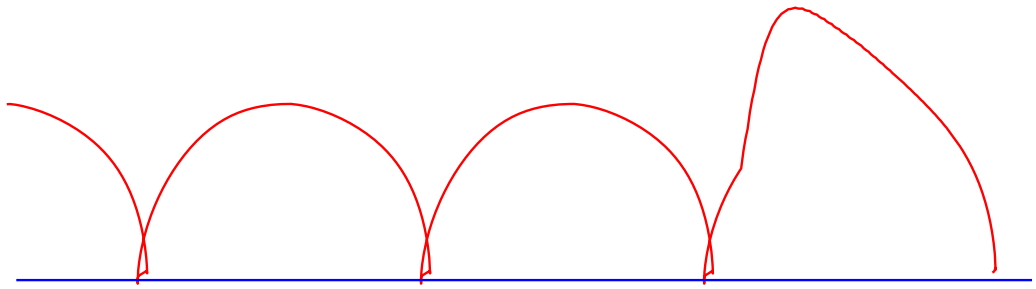


Figure 5.28: Trace of the right foot during one of the iterations of phase 2 of Scenario 4. *The leg reacts to the obstacle, even though it was not necessary to do so.*

right area and the rest of the lower area, the weights on the lower left area of the image are still too high, therefore causing erroneous reactions to objects which should not have any impact on the behaviour.

As this was a moderately serious problem, a more detailed analysis was done on what was causing it and how to improve the system so that it was more robust to these kinds of situations. In order address this problem, it was hypothesised that forgetfulness would be beneficial, as the cause of the problem was the fact that the system had learned an association that proved to be no longer valid after a few trials. Without forgetfulness, this association will be permanent and will cause disturbance to the normal behaviour of the robot. With forgetfulness present, the weights of this association would decrease and eventually would no longer cause problems. These details, along with the proposed extension to the ISO3 algorithm are described fully in sections 3.4.3 and 3.4.4.

5.2.4 Closed Loop Experiments using Forgetful ISO3 learning

In section 3.4.4, an alternative implementation of the learning algorithm, named Forgetful ISO3, was presented and successfully tested with artificial inputs. This implementation was different to the one used in the previous experiments in the fact that it allows for weights to decrease, mostly in lower activity synapses. As it was expected to have a better performance, it was used as a replacement learning system and, in order to verify that it worked as expected, the same experiments described in the previous section were performed again.

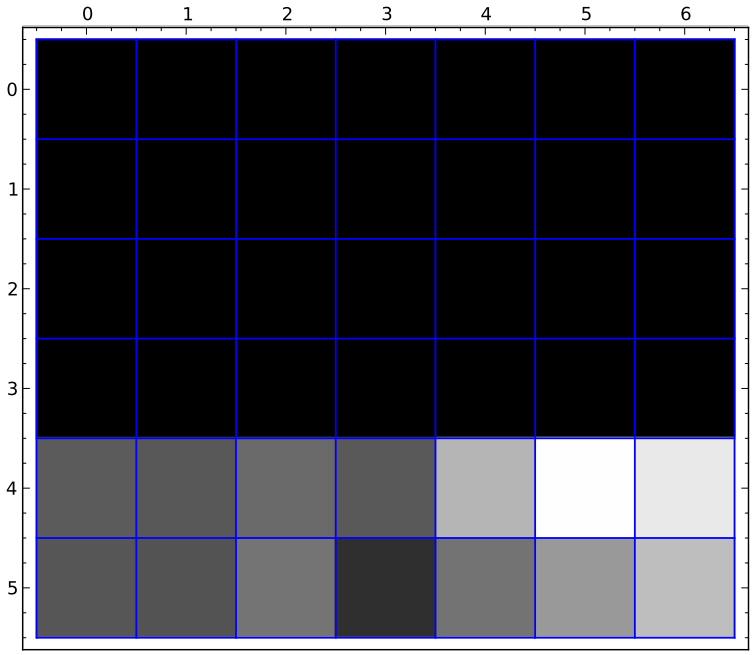


Figure 5.29: Plot of the sum of the weights for each of the blocks at the end of phase 1 of Scenario 4. *In this case, the areas to the right of the image have stronger weights, but the rest of the areas still have too high weights due to the training examples where the object was placed at the centre. Black represents 0 weight and white represents the higher of the weights.*

For the first scenario, in which the object is approached at its centre, the system is shown to successfully learn how to climb the obstacle. The trace of the right foot can be seen in figure 5.31. In this case, the weights have a similar distribution to the one already observed in the previous section, as the entire lower area is active during the reflex events (figure 5.33).

For the second scenario, this system also performed as expected and, again, the system was able to learn successfully how to climb the obstacle without generating a reflex event (figures 5.34 and 5.35). The weight distribution (figure 5.36) is similar to the one obtained in the previous section, but it can already be seen that some of the lower activity synapses, that were previously barely visible but still present, have been suppressed.

Since the system was successful on both of these scenarios, it shows that the changes performed did not have a negative impact on the basic behaviours, when compared with the previous algorithm. In order to verify whether the system has improved on

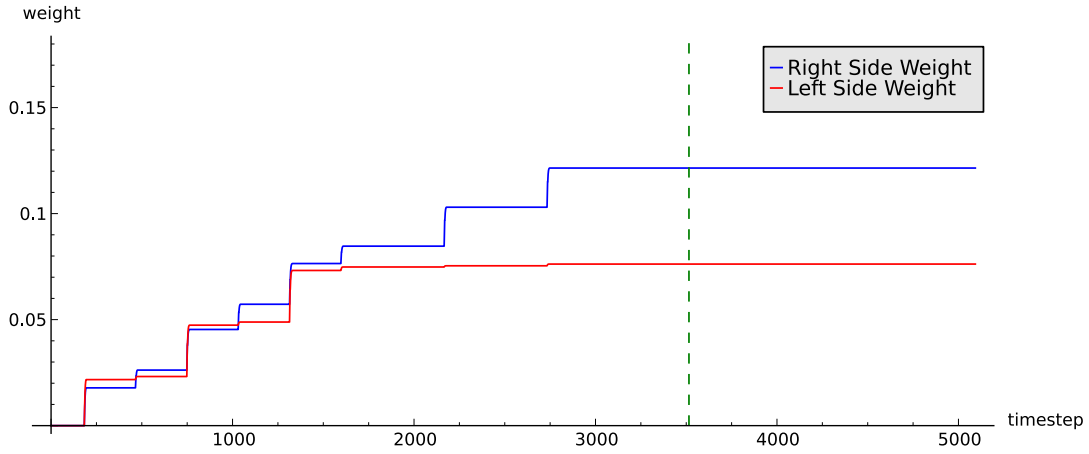


***Figure 5.30: Trace of the weights for Scenario 4 for the two sides of the image**. In this case it can be seen that in the first phase, the weights from the right side of the screen increased more than the weights on the left, but after this phase finishes, the left side weights already have a considerable value which is kept in phase 2 due to the absence of reflexes.*

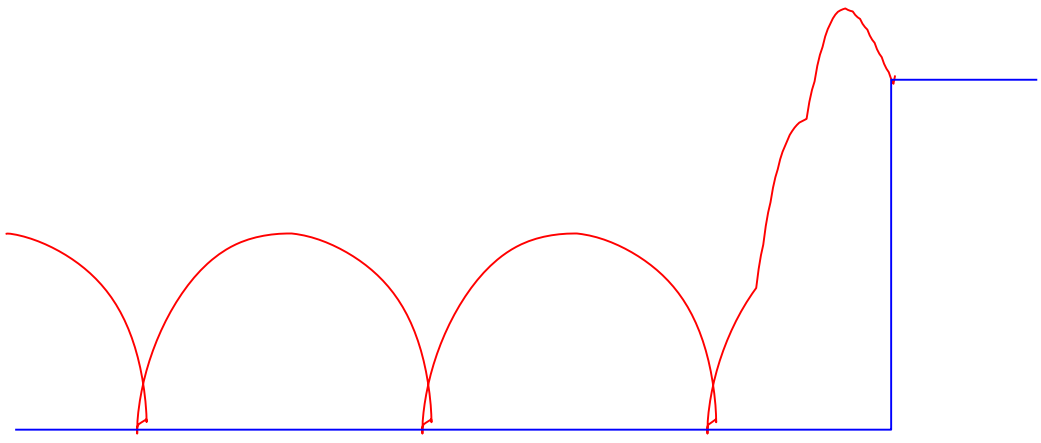


Figure 5.31: Trace of the right foot after learning is successful in overcoming the obstacle from figure 5.16.

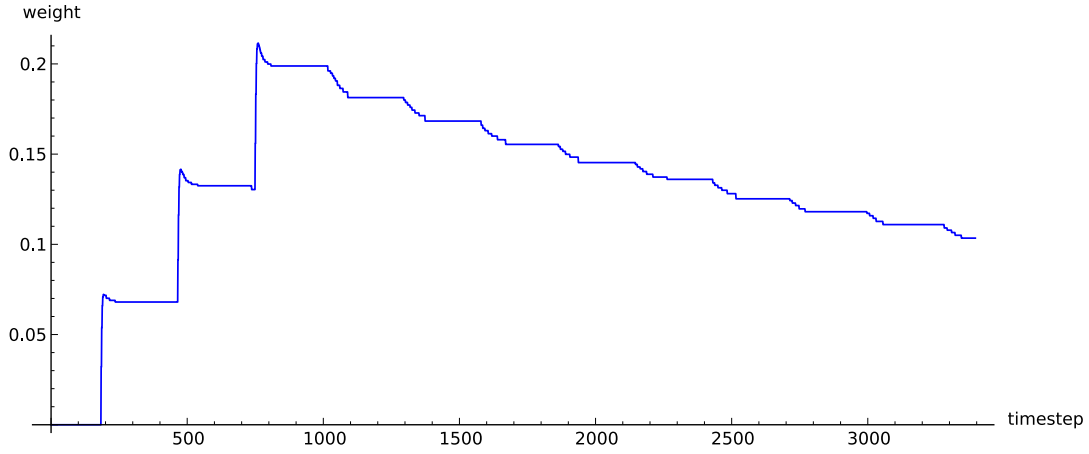


Figure 5.32: Trace of the weights for Scenario 1 using Forgetful ISO3. *In this case, after a weight high enough to overcome the obstacles was achieved, The forgetfulness factor starts having its influence and it can be seen that the weights start decreasing. If the simulation had continued for longer, reflex actions would be actioned again because the weight would decrease to a level below the one needed to overcome the obstacle. This happens especially because there are many input paths have similarly high weights, which causes them to suppress each other in a consistent manner.*

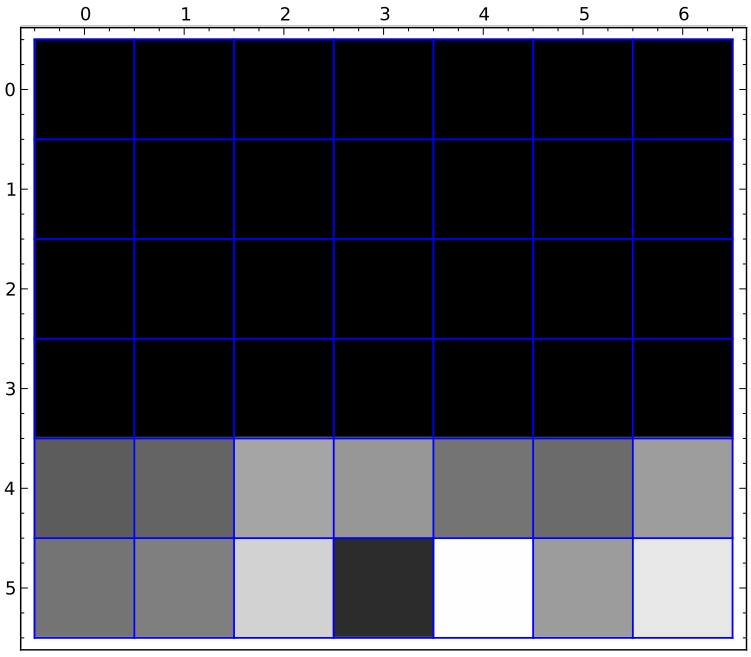


Figure 5.33: Plot of the sum of the weights for each of the blocks at the end of Scenario 1 using Forgetful ISO3. *This shows the areas in the image which were considered to be more important to predict the obstacle. Black represents 0 weight and white represents the higher of the weights.*

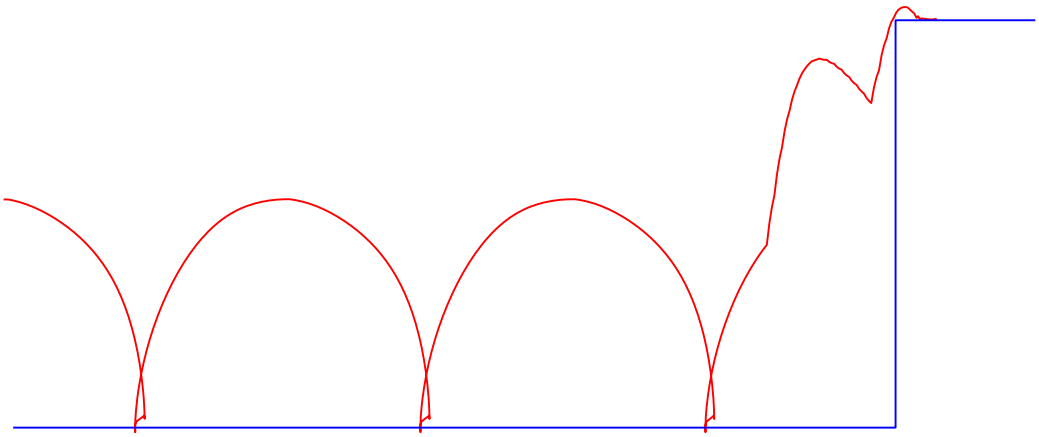


Figure 5.34: Trace of the right foot after learning is successful in overcoming the obstacle in Scenario 2 *(where the object was placed on the right).*

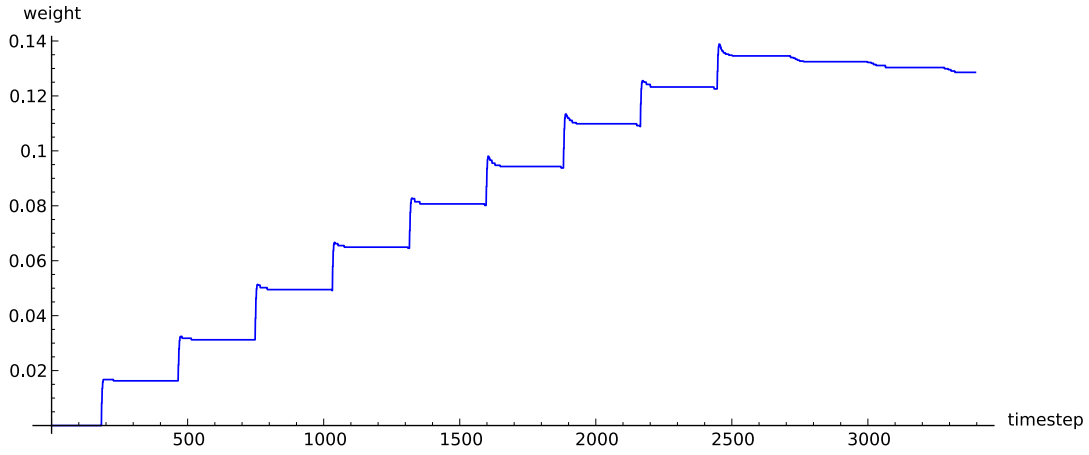


Figure 5.35: Trace of the weights for Scenario 2 using Forgetful ISO3*. The weights take longer to reach point at which no more reflexes take place due to the lower input activity present.*

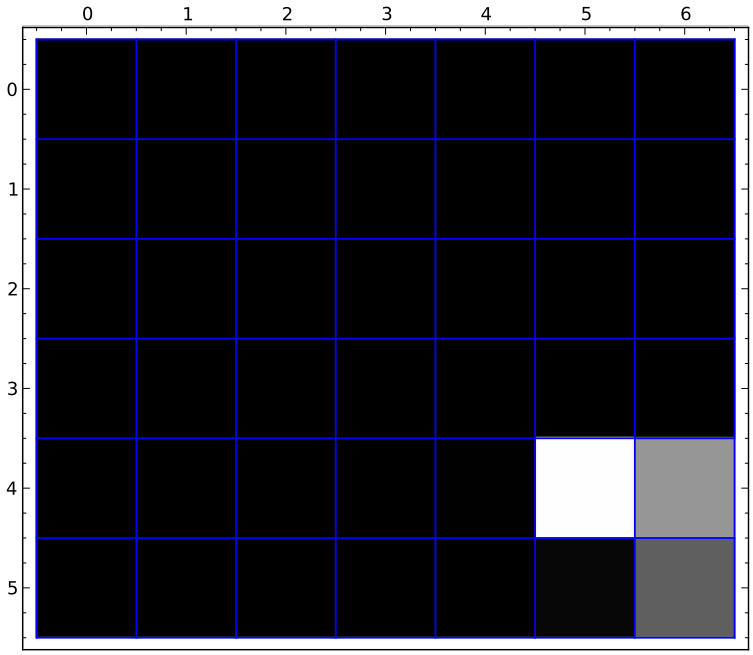


Figure 5.36: Plot of the sum of the weights for each of the blocks at the end of Scenario 2 using Forgetful ISO3. *This shows the areas in the image which were considered to be more important to predict the obstacle. Black represents 0 weight and white represents the higher of the weights.*

the problems observed in the previous system, it was also tested on the third and fourth scenarios.

In the third scenario, the system successfully adjusted its weights to climb the obstacle in the first and second phases as expected. After the second phase though, the weight distribution (figure 5.37) was clearly different from the one obtained before. During the second phase, the system was able to almost completely suppress the weights relating to the activity from the lower left area of the image and only the most significant areas were left with significant weights (figure 5.39). In the third phase, this system was confirmed to have resolved the problem detected in the previous section, as can be shown from the trace of its right foot in figure 5.38. The right leg does not change its trajectory when the system is faced with the obstacle on the left, as it has correctly considered it to be in an area of no importance.

In the fourth scenario, this system had the potential to not behave as correctly as in the third one, as the activity from the lower left never fully disappears during learning. After this scenario was tested, observing the foot trace from figure 5.40, it was verified

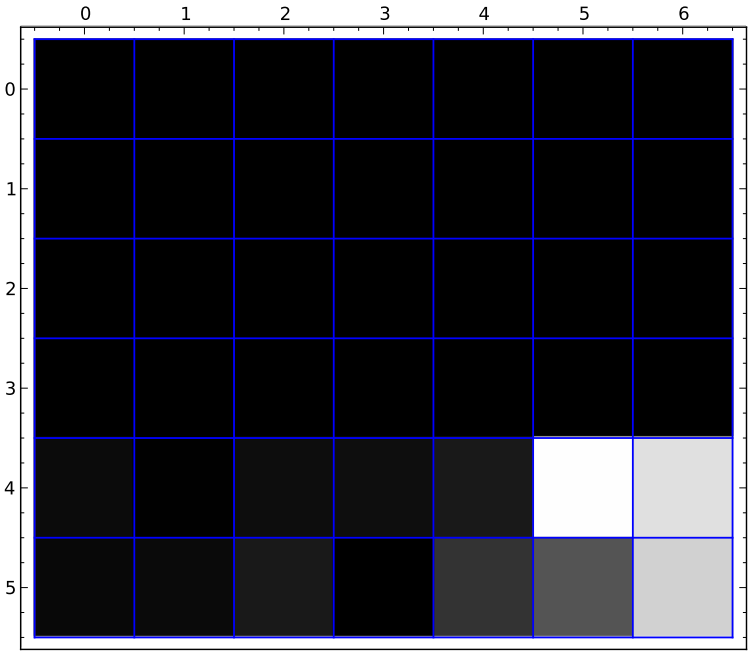


Figure 5.37: Plot of the sum of the weights for each of the blocks at the end of phase 2 of Scenario 3 using Forgetful ISO3. *In this case, the areas in the lower right of the image are the only ones that still have weights as the rest have been suppressed, unlike the results obtained with the Spiking ISO3 algorithm. Black represents 0 weight and white represents the higher of the weights.*

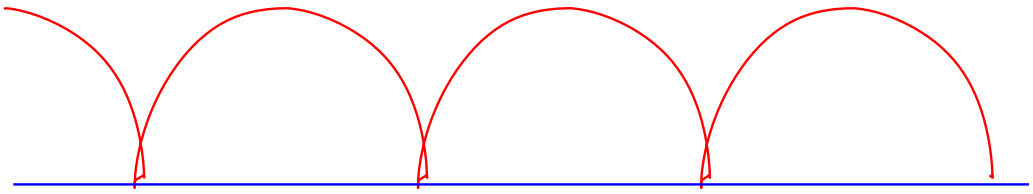


Figure 5.38: Trace of the right foot during one of the iterations of phase 3 of Scenario 3 using Forgetful ISO3. *The system correctly ignores the obstacle and the leg performs normal walking cycles.*

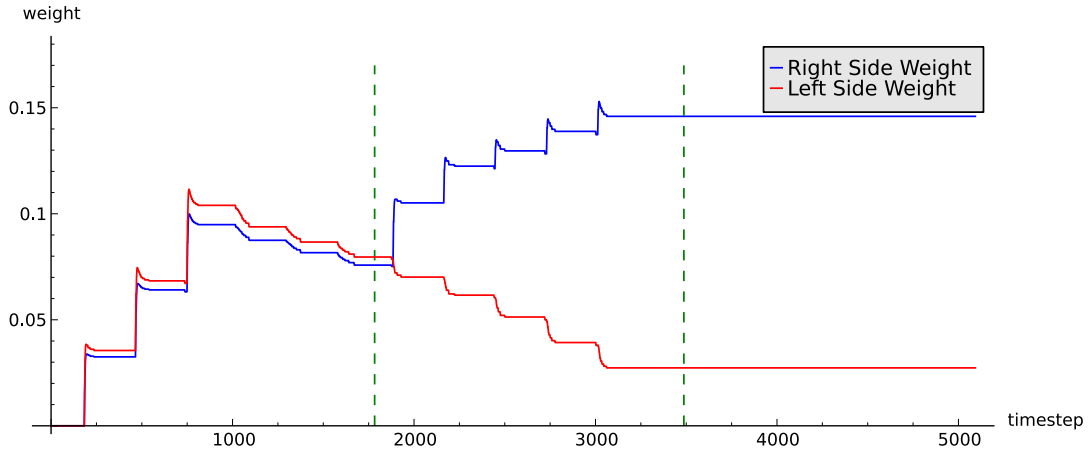


Figure 5.39: Trace of the weights for Scenario 3 using Forgetful ISO3 for the two sides of the image. *The right side weights suppressed the left side weights to a point where they no longer affect the output, so they no longer have an suppressive effect on the right side weights.*

that the system still performed correctly without any deviation observable in the right leg. Analysing the weight distribution in figures 5.41 and 5.42, it can be seen that the system was still able to suppress the weights on the lower left area of the image to a point where they do not have any significant influence on the leg trajectory.

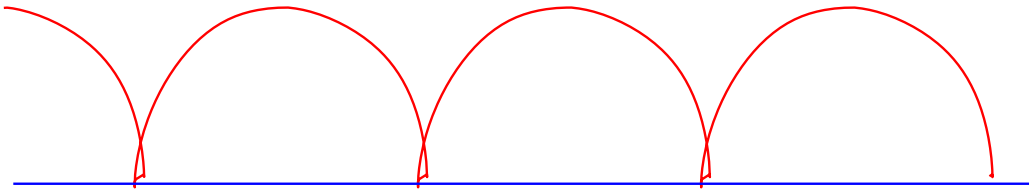


Figure 5.40: Trace of the right foot during one of the iterations of phase 2 of Scenario 4 using Forgetful ISO3. *The system correctly ignores the obstacle and the leg performs normal walking cycles.*

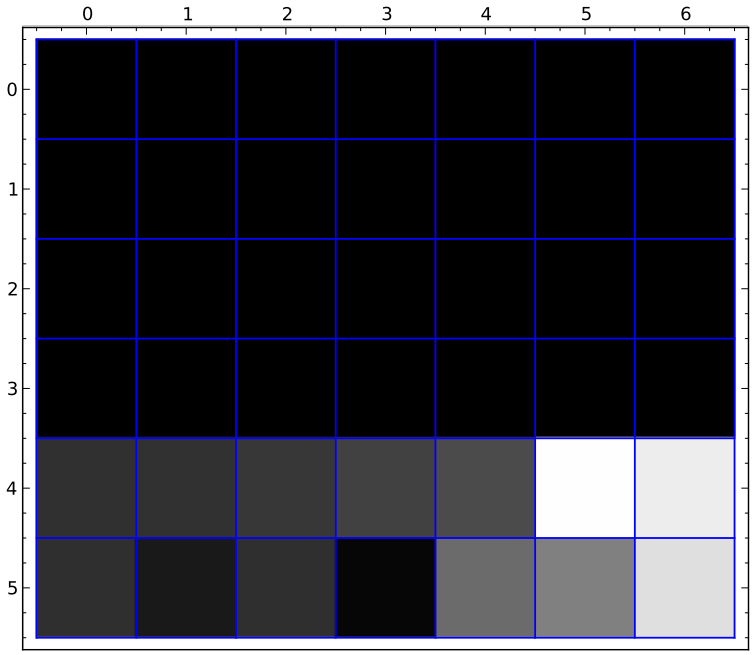


Figure 5.41: Plot of the sum of the weights for each of the blocks at the end of phase 1 of Scenario 4 using Forgetful ISO3. *In this case, the areas to the right of the image gain importance and the rest is suppressed to levels low enough not to cause an effect on the leg control. Black represents 0 weight and white represents the higher of the weights.*

5.3 Summary

In this chapter, the integration of all the systems (stereo vision, obstacle detection, learning and walking control) was described. Several test scenarios were used for evaluation of the system performance. These consisted of presenting obstacles in different positions of the field of view and verifying whether the robot managed to learn how to correctly surpass them. The system was mostly successful on simple scenarios, but had problems on more complex ones. After an analysis was made and the problem identified, changes were introduced to the the learning algorithm in order to improve its performance. These changes consisted in adding the capability for the system to "forget" and at the same time for the synapses to suppress each other. The tests were executed again and the modified algorithm was confirmed to have improved on the performance of the original system by correctly adjusting to all the scenarios tested.

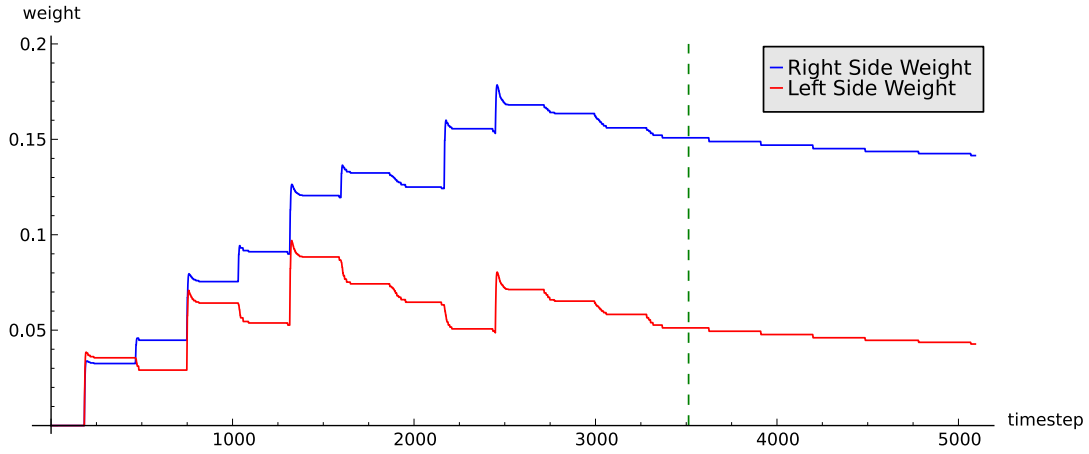


Figure 5.42: Trace of the weights for Scenario 4 using Forgetful ISO3 for the two sides of the image. *The right side weights suppressed the left side weights to a point where their activity is minor but still present, which still elicits the competition process thus leading to weight decrease in the later stages.*

Chapter 6
Summary and Conclusions

6.1 Summary

Here are summarised the activities undertaken and results of the several components of the project addressed in this thesis.

Learning

In Chapter 3, two different types of learning were implemented and tested, the first was a spiking implementation of the ISO3 learning (Spiking ISO3) and the second (Forgetful ISO3) had an added term in the weight update rule, which allowed for the weaker synapses to be suppressed. Using ISO3 as a base instead of normal ISO gave a higher stability and resilience to the algorithm due to this system already having a mechanism to avoid uncontrolled auto-correlation effects.

The Spiking ISO3 version appeared to work correctly in all the simple simulations performed, which had only one predictive input and no noise, but when this version was tested in the 3D simulations, where the system was used with a high number of different and slightly noisy predictive inputs, it was verified that the system learnt how to associate certain inputs with the reflex event leading to an increase of its weights, but was not able to detect that these associations were purely coincidental. This inability to "forget" wrong associations affected the walking controller in inappropriate ways, causing it to alter its gait at wrong times.

The Forgetful ISO3 version was created in order to attempt to solve the problem described above. The extra terms in its weight update rule were partially inspired in the Oja rule, which defines that the amount of weight decrease should be proportional to the amount of output activity (in this case it was changed to the amount of variation in the output activity) and also took inspiration from synaptic suppression

where stronger synapses inhibit weaker ones. After testing this on an open loop simulation modelling the conditions where the problem was detected, it showed a good performance.

Vision

The vision system implemented consisted in an obstacle detection system which used depth data as its input and had several outputs, one for each voxel in the scene. These outputs emitted spikes when an obstacle was detected in the corresponding voxel.

The source of depth data which was planned to be used was a depth from motion system which received its data from an artificial retina. After some preliminary tests where data from the artificial retina was recorded and fed to a model of the depth from motion chip, some potential problems were detected which prompted further investigations. The first problem, and the most visible in the tests performed, was the interaction of the camera noise characteristics and the error detection system in the algorithm. As the camera produced a noisy output, it interfered with the overzealous error detection system, which would then discard most of the measurements taken. In an attempt to surpass this problem, the data from the camera was filtered and simulations were performed. This time, although the results improved, many measurements were still being discarded.

The other problem found was related to the small movements occurring in the camera at the time of the testing, which had an impact on the timing of the edges detected. A mathematical analysis was then performed in order to verify the effects of the wobbling movements which were going to be inserted by the robot locomotion. This showed that the system would not cope at all even with slight movements, as the measurements would be very distorted.

Both of these problems indicated that the system would need serious changes before it could be used for this application. So instead of using it, the obstacle detection system made use of a stereo vision setup. This system used two normal cameras to acquire the scene, calculated its disparity using an off the shelf system which was fed to a layer of disparity selective integrate and fire neurons. These neurons were then connected with excitatory and inhibitory synapses to a second layer of integrate and

fire neurons, these with an habituation mechanism in order to detect novelty. In the preliminary tests this system was robust to the predicted movement of the robot and was able to mostly ignore the ground as it was approaching in certain phases of the walking cycle, whilst being able to detect incoming obstacles.

Integration

When all the systems were integrated together, several simulations were run using different test scenarios which consisted of placing the obstacles in different positions for several iterations and analyse whether the robot managed to adapt its gait to allow the climbing of the obstacles without a reflex action taking place. In most of the tests, the system managed to adapt the gait to the presence of the obstacles, but some problems were detected in some of the tests. These problems were investigated and linked to the learning algorithm.

As described above, the learning algorithm was changed and the Forgetful ISO3 learning was implemented. This version, which had been successful in the preliminary results, was tested using the same scenarios and it was observed that the performance of the system had improved, especially in the problematic cases in which the Spiking ISO3 was not able to adapt correctly. In these cases, the Forgetful ISO3 managed to adapt correctly to the terrain therefore showing a better capacity for handling noisy inputs and coincidental correlations in the predictive events while still maintaining the same level of performance as the Spiking ISO3 in the rest of the scenarios tested.

6.2 Future Work

Even though the experiments were performed in a reasonably accurate simulator, deploying the systems in an actual physical context would be even more useful, as the sources of noise, faults, and other potential imperfections in the hardware implementation would give a greater insight into how robust the system is and how it could be improved.

Another activity which would be interesting to perform would be to integrate this system with a more complex low level control system like SCASM or, even more

interesting, a spiking neuron implementation of the actual biological mechanism described in [100].

In order to save more power, the full synapse elimination mechanism [74] could be implemented, so that the synapses which were left with no weights could be turned off or be used by other components of the system needing more computing elements.

One question which would be important to answer is, as the learning system software implementation was performed in a way which allows for an easy translation to hardware, whether an analogue VLSI implementation of this system would be more power efficient than the traditional approach of having a generic processor or field programmable gate array (FPGA) control the robot actions.

6.3 Conclusions

This project explores the suggestion that biologically inspired control and processing systems can improve hexapod walking behaviour on uneven surfaces by using a biologically inspired unsupervised learning algorithm to use sensor readings to adapt its gait. Through the implementations and integration of multiple biologically inspired systems it was shown that it is already possible to apply them with positive outcomes in this type of scenario.

At Bielefeld University, there is a very active community studying insect locomotion and building robots based on those studies. HECTOR is the most recent robot arising from that collaboration, which uses WALKNET as its control system.

While WALKNET and SCASM (based on Cruse studies) provide very good reflex based distributed walking control systems, they are completely reliant upon previous configuration and do not change their behaviour with experience. This is where the learning system explored in this thesis would be able to help to increase the performance of the system without requiring pre-tuning of the action parameters by allowing another layer of sensory fusion to evolve by itself instead.

It was also verified that the addition of synaptic competition has an important impact, especially in systems which work with imperfect data or in situations in which some correlations learnt turn out not to be correct. In these cases, the ability to suppress

the weak synapses created by such erroneous learning events allows for the correct behaviour to be restored.

While at first sight the forgetful ISO3 algorithm might appear to adopt a similar strategy to a winner takes all system, it has some fundamental differences. In the presence of a reflex signal, all predictive inputs which correlate with it will have their weights increased and not just the strongest one. This keeps the system from relying on any particular input early on. In this system, the pruning and differentiation occurs particularly when some inputs correlate with the reflex signal, but not others, in which case, the latter are strongly suppressed by the former. It will continue to occur, in a smaller way after the reflex signal stops being activated, especially if there is already a goog differentiation at that moment. In the situation where there are several predictive inputs with similar weights, all being equally active but at slightly different times, the system will exhibit a general weight decrease mimicking forgetfulness, as these inputs activity will suppress each other in similar amounts. This will eventually lead to the reappearance of a reflex action, which might then elicit a stronger discrimination between them.

Unlike the original ISO3 algorithm, if a particular input stops being correlated with the situations being avoided, its weight will decrease in favour of other inputs with stronger correlation. This helps deal both with noisy sensor inputs and with changes in circumstances.

Some systems still require extra work in order to make them applicable to the real world like the depth from motion system implementation, which requires a higher robustness against movement disturbances and input noise before it can be used in an application where the cameras do not have a constant unidirectional movement.

References

[1] J. Marescaux, J. Leroy, M. Gagner, F. Rubino, D. Mutter, M. Vix, S. E. Butner, and M. K. Smith, "Transatlantic robot-assisted telesurgery," *Nature*, vol. 413, pp. 379–380, Sept. 2001.

[2] J. Matijevic, "Autonomous navigation and the sojourner microrover," *Science*, vol. 280, pp. 454–455, Apr. 1998.

[3] K. Sanderson, "Mars rover spirit (2003–10)," *Nature*, vol. 463, p. 4, 2010.

[4] R. Lindemann, D. Bickler, B. Harrington, G. Ortiz, and C. Voothees, "Mars exploration rover mobility development," *Robotics Automation Magazine, IEEE*, vol. 13, pp. 19 –26, June 2006.

[5] C. Baker, A. Morris, D. Ferguson, S. Thayer, C. Whittaker, Z. Omohundro, C. Reverte, W. Whittaker, D. Hahnel, and S. Thrun, "A campaign in autonomous mine mapping," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 2, pp. 2004 – 2009 Vol.2, May 2004.

[6] R. Murphy, J. Kravitz, K. Peligren, J. Milward, and J. Stanway, "Preliminary report: Rescue robot at crandall canyon, utah, mine disaster," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2205 –2206, May 2008.

[7] K. G. Pearson and J. F. Iles, "Nervous mechanisms underlying intersegmental co-ordination of leg movements during walking in the cockroach," *Journal of Experimental Biology*, vol. 58, pp. 725–744, Jan. 1973.

[8] T. Akay, S. Haehn, J. Schmitz, and A. Buschges, "Signals from load sensors underlie interjoint coordination during stepping movements of the stick insect leg," *J Neurophysiol*, vol. 92, pp. 42–51, July 2004.

[9] H. Cruse, V. Dürr, and J. Schmitz, "Insect walking is based on a decentralized architecture revealing a simple and robust controller," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, pp. 221–250, Jan. 2007.

[10] H. Cruse, "Coordination of leg movement in walking animals," in *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, (Paris, France), pp. 105–119, MIT Press, 1990.

[11] W. Lewinger and R. Quinn, "BILL-LEGS: low computation emergent gait system for small mobile robots," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 251–256, 2008.

[12] P. Arena, L. Fortuna, M. Frasca, L. Patane, and M. Pollino, "An autonomous mini-hexapod robot controlled through a CNN-based CPG VLSI chip," in *Cellular Neural Networks and Their Applications, 2006. CNNA '06. 10th International Workshop on*, pp. 1 –6, Aug. 2006.

[13] R. Siegwart and I. R. Nourbakhsh, *Introduction to autonomous mobile robots*. MIT press, 2004.

[14] B. Siciliano and O. Khatib, eds., *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[15] M. J. Spenko, G. C. Haynes, J. A. Saunders, M. R. Cutkosky, A. A. Rizzi, R. J. Full, and D. E. Koditschek, "Biologically inspired climbing with a hexapedal robot," *Journal of Field Robotics*, vol. 25, no. 4-5, pp. 223–242, 2008.

[16] M. Kaneko, A. Mizuno, and K. Harada, "Torque distribution for achieving a hugging walk," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3, pp. 2613 – 2618 vol.3, 2002.

[17] V. L. Krishnan, P. M. Pathak, S. C. Jain, and A. K. Samantaray, "Reconfiguration of four-legged walking robot for actuator faults," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 226, pp. 11–26, Jan. 2012.

[18] J.-M. Yang and J.-H. Kim, "A strategy of optimal fault tolerant gait for the hexapod robot in crab walking," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 2, pp. 1695 –1700 vol.2, May 1998.

[19] Y.-J. Lee and S. Hirose, "Three-legged walking for fault tolerant locomotion of a quadruped robot with demining mission," in *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, vol. 2, pp. 973 –978 vol.2, 2000.

[20] J.-M. Yang, "Fault-tolerant gait generation for locked joint failures," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 3, pp. 2237 – 2242 vol.3, Oct. 2003.

[21] "Toshiba develops tetrapod robot for tokyo electric power plant fukushima no.1 nuclear power plant."
http://www.toshiba.co.jp/about/press/2012_11/pr2101.htm, Nov. 2012.

[22] K. Kamikawa, T. Arai, K. Inoue, and Y. Mae, "Omni-directional gait of multi-legged rescue robot," in *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04*, vol. 3, pp. 2171 – 2176 Vol.3, May 2004.

[23] B. H. Wilcox, T. Litwin, J. Biesiadecki, J. Matthews, M. Heverly, J. Morrison, J. Townsend, N. Ahmad, A. Sirota, and B. Cooper, "Athlete: A cargo handling and manipulation robot for the moon," *Journal of Field Robotics*, vol. 24, no. 5, p. 421–434, 2007.

[24] "JPL robotics: System: The ATHLETE rover."
http://www-robotics.jpl.nasa.gov/systems/system.cfm?System=11.

[25] "Honda ASIMO walking stairs - wikimedia commons." http://commons.wikimedia.org/wiki/File:Honda_ASIMO_Walking_Stairs.JPG.

[26] M. H. Dickinson, C. T. Farley, R. J. Full, M. A. R. Koehl, R. Kram, and S. Lehman, "How animals move: An integrative view," *Science*, vol. 288, pp. 100–106, Apr. 2000.

[27] R. McGhee and A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Biosciences*, vol. 3, pp. 331–351, Aug. 1968.

[28] S. Kajita and B. Espiau, "Legged robots," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), pp. 361–389, Springer Berlin Heidelberg, Jan. 2008.

[29] M. Buehler, R. Playter, and M. Raibert, "Robots step outside," in *Int. Symp. Adaptive Motion of Animals and Machines (AMAM), Ilmenau, Germany*, p. 1–4, 2005.

[30] G. Nelson, K. Blankespoor, and M. Raibert, "Walking BigDog: insights and challenges from legged robotics," *Journal of Biomechanics*, vol. 39, Supplement 1, p. S360, 2006.

[31] M. Raibert, K. Blankespoor, G. Nelson, R. Playter, *et al.*, "Bigdog, the rough-terrain quadruped robot," *Proceedings of the 17th International Federation of Automation Control*, p. 10822–10825, 2008.

[32] "Boston dynamics BigDog - military factory." http://www.militaryfactory.com/armor/detail.asp?armor_id=184.

[33] M. Schilling, H. Cruse, and P. Arena, "Hexapod walking: an expansion to walknet dealing with leg amputations and force oscillations," *Biological Cybernetics*, vol. 96, pp. 323–340, Mar. 2007.

[34] A. Schneider, J. Paskarbeit, M. Schäffersmann, and J. Schmitz, "Biomechatronics for embodied intelligence of an insectoid robot," in *Intelligent Robotics and Applications* (S. Jeschke, H. Liu, and D. Schilberg, eds.), no. 7102 in Lecture Notes in Computer Science, pp. 1–11, Springer Berlin Heidelberg, Jan. 2011.

[35] M. Frik, M. Guddat, D. C. Losch, and M. Karatas, "Terrain adaptive control of the walking machine tarry II," in *Proc. European Mechanics Colloquium, Euromech*, vol. 375, p. 108–115, 1998.

[36] D. Kingsley, R. Quinn, and R. Ritzmann, "A cockroach inspired robot with artificial muscles," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 1837 –1842, Oct. 2006.

[37] B. Jakimovski, *Cognitive Systems Monographs: Biologically Inspired Approaches for Locomotion, Anomaly Detection and Reconfiguration for Walking Robots*. Springer, Jan. 2011.

[38] W. A. Lewinger, B. L. Rutter, M. Blümel, A. Büschges, and R. D. Quinn, "Sensory coupled action switching modules (SCASM) generate robust, adaptive stepping in legged robots," in *CLAWAR*, vol. 2006, p. 9th, 2006.

[39] E. Marder and D. Bucher, "Central pattern generators and the control of rhythmic movements," *Current Biology*, vol. 11, pp. R986–R996, Nov. 2001.

[40] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, pp. 642–653, May 2008.

[41] U. Bässler and A. Büschges, "Pattern generation for stick insect walking movements—multisensory control of a locomotor program," *Brain Research Reviews*, vol. 27, pp. 65–88, June 1998.

[42] H. Cruse, T. Kindermann, M. Schumm, J. Dean, and J. Schmitz, "Walknet—a biologically inspired network to control six-legged walking," *Neural Networks*, vol. 11, pp. 1435–1447, Oct. 1998.

[43] W. Lewinger, C. Harley, R. Ritzmann, M. Branicky, and R. Quinn, "Insect-like antennal sensing for climbing and tunneling behavior in a biologically-inspired mobile robot," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 4176–4181, 2005.

[44] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120 dB 15 µs latency asynchronous temporal contrast vision sensor," *Solid-State Circuits, IEEE Journal of*, vol. 43, no. 2, pp. 566–576, 2008.

[45] "Silicon retina." http://siliconretina.ini.uzh.ch/wiki/index.php.

[46] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip i: Outer and inner retina models," *IEEE Transactions on Biomedical Engineering*, vol. 51, pp. 657–666, Apr. 2004.

[47] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip II: testing and results," *IEEE Transactions on Biomedical Engineering*, vol. 51, pp. 667–675, Apr. 2004.

[48] F. Wörgötter and B. Porr, "Reinforcement learning," *Scholarpedia*, vol. 3, no. 3, p. 1448, 2008.

[49] D. O. Hebb, *The organization of behavior; a neuropsychological theory.* New York: Wiley, 1949.

[50] W. Gerstner, "Hebbian learning and plasticity," 2011.

[51] Y. Niv, "Reinforcement learning in the brain," *Journal of Mathematical Psychology*, vol. 53, pp. 139–154, June 2009.

[52] L. Abbott, S. Nelson, *et al.*, "Synaptic plasticity: taming the beast," *Nature neuroscience*, vol. 3, p. 1178–1183, 2000.

[53] P. F. Verschure, B. J. Kröse, and R. Pfeifer, "Distributed adaptive control: The self-organization of structured behavior," *Robotics and Autonomous Systems*, vol. 9, no. 3, pp. 181–196, 1992.

[54] B. Porr, C. v. Ferber, and F. Wörgötter, "ISO learning approximates a solution to the inverse-controller problem in an unsupervised behavioral paradigm," *Neural Computation*, vol. 15, pp. 865–884, Apr. 2003.

[55] B. Porr and F. Wörgötter, "Isotropic sequence order learning," *Neural Comp.*, vol. 15, pp. 831–864, Apr. 2003.

[56] E. Oja, "Simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, pp. 267–273, Nov. 1982.

[57] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex," *The Journal of Neuroscience*, vol. 2, pp. 32–48, Jan. 1982.

[58] P. Easton and P. E. Gordon, "Stabilization of hebbian neural nets by inhibitory learning," *Biological Cybernetics*, vol. 51, pp. 1–9, Oct. 1984.

[59] U. Mortensen and C. Nachtigall, "Visual channels, hebbian assemblies and the effect of hebb's rule," *Biological Cybernetics*, vol. 82, pp. 401–413, Apr. 2000.

[60] W. Gerstner and W. M. Kistler, "Mathematical formulations of hebbian learning," *Biological Cybernetics*, vol. 87, pp. 404–415, Dec. 2002.

[61] K. D. Miller and D. J. C. MacKay, "The role of constraints in hebbian learning," *Neural Computation*, vol. 6, pp. 100–126, Jan. 1994.

[62] B. Kosko, "Differential hebbian learning," *AIP Conference Proceedings*, vol. 151, no. 1, pp. 277–282, 1986.

[63] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.

[64] B. Porr and F. Wörgötter, "Fast heterosynaptic learning in a robot food retrieval task inspired by the limbic system," *Biosystems*, vol. 89, no. 1-3, pp. 294–299, 2007.

[65] B. Porr and F. Wörgötter, "Strongly improved stability and faster convergence of temporal sequence learning by using input correlations only," *Neural Comp.*, vol. 18, pp. 1380–1412, June 2006.

[66] B. Porr, T. Kulvicius, and F. Wörgötter, "Improved stability and convergence with three factor learning," *Neurocomputing*, vol. 70, pp. 2005–2008, June 2007.

[67] B. Porr and F. Wörgötter, "Learning with relevance: Using a third factor to stabilize hebbian learning," *Neural Computation*, vol. 19, pp. 2694–2719, Oct. 2007.

[68] C. Kolodziejski, B. Porr, and F. Wörgötter, "Mathematical properties of neuronal TD-rules and differential hebbian learning: a comparison," *Biological Cybernetics*, vol. 98, pp. 259–272, Mar. 2008.

[69] B. Porr and F. Worgotter, "Learning a forward model of a reflex," *Advances in Neural Information Processing Systems*, pp. 1555–1562, 2003.

[70] G. Bi and M. Poo, "Synaptic modification by correlated activity: Hebb's postulate revisited," *Annual Review of Neuroscience*, vol. 24, pp. 139–166, Mar. 2001.

[71] J. Sjöström and W. Gerstner, "Spike-timing dependent plasticity," *Scholarpedia*, vol. 5, no. 2, p. 1362, 2010.

[72] W. Gerstner and W. Kistler, *Spiking Neuron Models: An Introduction*. New York, NY, USA: Cambridge University Press, 2002.

[73] P. Dayan and L. F. Abbott, *Theoretical neuroscience*, vol. 83. Citeseer, 2001.

[74] J. W. Lichtman and H. Colman, "Synapse elimination and indelible memory," *Neuron*, vol. 25, pp. 269–278, Feb. 2000.

[75] M. A. Arbib, G. Metta, and P. v. d. Smagt, "Neurorobotics: From vision to action," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), pp. 1453–1480, Springer Berlin Heidelberg, Jan. 2008.

[76] I. P. Howard and B. J. Rogers, *Binocular vision and stereopsis*. Oxford psychology series, No. 29., New York, NY, US: Oxford University Press, 1995.

[77] W. Harris, "Evolution of binocular and stereoscopic vision in man and other animals," *BMJ*, vol. 2, pp. 297–301, Aug. 1953.

[78] K. Karl, "Behavioural - analytical studies of the role of head movements in depth perception in insects, birds and mammals," *Behavioural Processes*, vol. 64, pp. 1–12, Aug. 2003.

[79] J. W. Nadler, D. E. Angelaki, and G. C. DeAngelis, "A neural representation of depth from motion parallax in macaque visual cortex," *Nature*, vol. 452, pp. 642–645, Apr. 2008.

[80] F. Wörgötter, A. Cozzi, and V. Gerdes, "A parallel noise-robust algorithm to recover depth information from radial flow fields," *Neural computation*, vol. 11, no. 2, pp. 381–416, 1999.

[81] M. A. Dahlem and F. Wörgötter, "Rotation-invariant optical flow by gaze-depended retino-cortical mapping," in *Biologically Motivated Computer Vision* (H. H. Bülthoff, C. Wallraven, S.-W. Lee, and T. A. Poggio, eds.), vol. 2525, pp. 137–145, Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

[82] Z. Yang, K. L. Cameron, A. F. Murray, and V. Boonsobhak, "An adaptive visual neuronal model implementing competitive, temporally asymmetric hebbian learning," *International Journal of Neural Systems*, vol. 16, pp. 151–162, June 2006. PMID: 17044237.

[83] Z. Yang and A. F. Murray, "An artificial early visual model adopting spike-timing-dependent plasticity," *Neurocomputing*, vol. 69, pp. 1904–1911, Oct. 2006.

[84] Z. Yang, A. F. Murray, F. Wörgötter, K. Cameron, and V. Boonsobhak, "A neuromorphic depth-from-motion vision model with STDP adaptation," *Neural Networks, IEEE Transactions on*, vol. 17, no. 2, pp. 482–495, 2006.

[85] K. Cameron and A. Murray, "Minimizing the effect of process mismatch in a neuromorphic system using spike-timing-dependent adaptation," *IEEE Transactions on Neural Networks*, vol. 19, pp. 899–913, May 2008.

[86] K. Cameron, V. Boonsobhak, A. Murray, and D. Renshaw, "Spike timing dependent plasticity (STDP) can ameliorate process variations in neuromorphic VLSI," *IEEE Transactions on Neural Networks*, vol. 16, pp. 1626–1637, Nov. 2005.

[87] V. Boonsobhak, K. Cameron, D. Renshaw, and A. Murray, "Compensating mismatch in a dedicated pixel array for moving edge detection," in *TENCON 2004. 2004 IEEE Region 10 Conference*, vol. D, pp. 278– 281 Vol. 4, IEEE, Nov. 2004.

[88] "Koala II." http://www.k-team.com/mobile-robotics-products/koala.

[89] N. Qian, "Binocular disparity and the perception of depth," *Neuron*, vol. 18, pp. 359–368, Mar. 1997.

[90] P. Rogister, R. Benosman, S.-H. Ieng, P. Lichtsteiner, and T. Delbruck, "Asynchronous event-based binocular stereo matching," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 23, pp. 347 –353, Feb. 2012.

[91] "Surveyor SRV-1 blackfin camera." http://www.surveyor.com/blackfin/.

[92] G. Bradski, "The opencv library," *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–124, 2000.

[93] D. Nair, "A guide to stereovision and 3D imaging :: NASA tech briefs." http://www.techbriefs.com/component/content/article/14925?start=1, 2012.

[94] H. Pashler and S. Yantis, *Sensation and perception*. New York, NY: Wiley, 2002.

[95] B. Blaesing and H. Cruse, "Stick insect locomotion in a complex environment: climbing over large gaps," *J Exp Biol*, vol. 207, pp. 1273–1286, Mar. 2004.

[96] K. Pearson and R. Franklin, "Characteristics of leg movements and patterns of coordination in locusts walking on rough terrain," *The International Journal of Robotics Research,*, pp. 101 –112 vol. 3:, June 1984.

[97] M. A. Lewis and L. S. Simo, "Elegant stepping: A model of visually triggered gait adaptation," *Connection Science*, vol. 11, no. 3, p. 331, 1999.

[98] "Stick insect - wikimedia commons." http://commons.wikimedia.org/wiki/File:Unknown.stick.insect.from.above.jpg.

[99] O. Michel, "Cyberbotics ltd. WebotsTM: professional mobile robot simulation," *International Journal of Advanced Robotic Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[100] R. E. Ritzmann and A. Buschges, "Adaptive motor behavior in insects," *Current Opinion in Neurobiology*, vol. 17, pp. 629–636, Dec. 2007.